SCODL
Programmer's Guide

Revision 5.40

June, 1989

imapro Corporation

COPYRIGHT

DISCLAIMER

Imapro reserves the right to make changes in specifications at any time and without notice. The information furnished by Imapro in this publication is believed to be accurate and reliable. However, no responsibility is assumed by Imapro for its use; nor for any infringements of patents or rights of third parties resulting from its use. No license is granted under any patents or patent rights of Imapro.

TRADEMARKS

SCODL, MVP+ are trademarks of Imapro Corp. PCR, TT200 are trademarks of Matrix Instruments Inc. QCR is a registered trademark of Matrix (P.E.I.) Inc. Mirage is a trademark of Zenographics Inc. Conjure is a trademark of Vision Control International Pty. Ltd. IBM is a registered trademark of International Business Machines.

imapro

Imapro Corp.
2400 St. Laurent Blvd.
Ottawa, Ontario
Canada K1G 5A4
(613) 738-3000
Fax (613) 738-5038

Imapro Corp.
2 Crosfield Avenue
West Nyack, NY 10994
U.S.A.
(914) 353-4701
Fax (914) 353-4715

Imapro Corp.
#180 - 2061 Business Center Dr.,
Irvine, CA 92715
U.S.A.
(714) 752-2043
Fax (714) 752-7544

Imapro Corp.
301 Highness Bldg.
3-7-26 Nishi - Shinjuku
Shinjuku-ku, Tokyo 160, Japan
(03) 342-3521
Fax (03) 342-3520

Revision 5.40 Manual Update

This manual describes the operation of the SCODL programming language corresponding to software version 5.40. This documentation represents a revision to the previous, "Version 5.32 March, 1988" manual. The changes and additions are as follows:

- Reformat and reorganize manual
- Add Opcode 224, subcommand 10 information (coordinate scaling command)
- Add Calcomp Plotmaster and Hewlett Packard LaserJet II descriptions to Appendix B, *Output Devices*

Table of Contents

SECTION 3
Describing Macros and Text

SECTION 4
Description of Opcodes

(The page numbering for Section 4 is listed on the bottom of the page according to the title or opcode description)

Appendices

Welcome to SCODL™!

Imapro's SCODL is an intermediate level language used to describe geometric objects within an x-y coordinate system. Once these objects are described in SCODL, they can then be translated to raster by sending the data to the Imapro MVP+ co-processor board. This procedure allows the data to be output at high resolution to raster-scan devices such as color ink-jet printers, thermal printers, and film recorders.

At last! SCODL offers you the capability to quickly output graphics to high resolution color output devices. And all easily performed using the Imapro MVP+ board and your personal microcomputer.

Before You Start . . .

The instructions in this manual assume that you are familiar with basic DOS operations. If you require learning information on DOS, refer to your DOS manual. Instructions for using Proscan assume you are using a Microsoft compatible mouse.

IMPORTANT: *This manual assumes that you have a working knowledge of DOS and its commands.*

What You Need To Use SCODL

You need the following to use SCODL:

- 286/386 IBM compatible computer
- MS DOS 3.2 or higher

How To Use this Manual

This manual provides users with instructions on how to program using SCODL.

Section Overview

The overview below provides a brief description of each section.

SECTION 1--*Overview* introduces the fundamentals concerning SCODL such as using opcodes, understanding the virtual screen coordinate system, and encoding data.

SECTION 2--*Describing Objects* explains how to describe objects in SCODL, determine coordinate data, select output image, rules on overlapping priorities, and information on RLC merging.

SECTION 3--*Describing Macros and Text* describes the contents of a SCODL image description. These include object descriptors, extended descriptors, and coordinate data.

SECTION 4--*Description of Opcodes*, explains how to use macros and specify text attributes. A complete character table of the Default Character Font is listed. Complete descriptions of all opcodes are listed in ascending order by opcode decimal number.

APPENDIX A--*The SCODL Status Field* describes the messages and commands in the SCODL Status Field of the Foreground MVP+ Display.

APPENDIX B--*Output Devices* describes the output devices that are supported by the Master Vector Processor Plus (MVP+) and SCODL.

APPENDIX C--*Coded Message Descriptions* describes MVP+ messages concerning processing errors, output configurations, and output status.

APPENDIX D--*Alphabetical List of Opcodes by Opcode Description* provides a listing of all opcodes alphabetically by their opcode name.

Typographical Format

This manual uses several typographical conventions. These include:

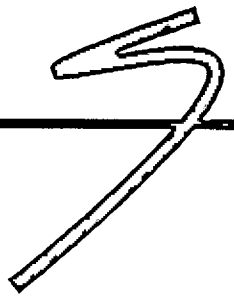
- **Scan Conversion Object Description Language** is abbreviated to **SCODL**.
- The term **Enter** means you must type in a command and press the <Enter> key.
- The term ^ means you must press the <Ctrl> key.
- During procedural steps, any commands to be entered and prompts are **bolded.**
- Filenames are written in upper case.
- All related manuals and sections have their titles italicized.

Related Manuals

You may need other manuals that are referenced in this manual to operate the system. These manuals are indicated below in alphabetical order:

- *DOS Technical Reference Manual*
- *Mitsubishi G650 Operator's Guide*
- *Programmer's Guide to SCODL*
- *Programmers and User Guide to the QCR*
- *QCS-450 Color Scanner User's Guide*
- *User's Guide to the PCR*
- *User's Guide to the TT200*

SECTION 1
Overview



This section contains:

- About SCODL
- Using Opcodes
- Virtual Screen Coordinate System
- Encoding Data

About SCODL

SCODL is an intermediate-level language used to describe geometric objects within an x-y coordinate system. Once these objects are described in SCODL, they can then be translated to raster by sending the SCODL data to the MVP+.

This enables the data to be output at high resolution to raster-scan devices such as color ink-jet printers, thermal printers, and film recorders.

Already Familiar with SCODL?

If you are already familiar with SCODL programming, read the following pages for a quick reference on the opcodes:

Section 4 provides you with the opcode priorities, an ascending list of all opcodes by their decimal values, the hexadecimal ASCII equivalent, the number of bytes, and complete descriptions of their use.

Appendix D provides an alphabetical reference list of all opcodes by their opcode description.

Using Opcodes

In order to initiate vector-to-raster processing, you must specify SCODL operation codes called *opcodes* to describe the image to be created. The first byte of an image description is allocated for an opcode, which is identified by a number between 0 and 255.

These opcodes may specify the following:

- Description of the object such as shape, color, arc width, and location of the coordinate data.
- Macro commands that group together a series of SCODL commands into a single command.
- Processing information including the dimensions of the image to be output, the repeat count, and the amount of memory to be allocated for the MVP+ internal processes.

Virtual Screen Coordinate System

All objects and dimensions are described in terms of an x-y coordinate grid system. This grid accumulates all the object descriptions in the image and is the basis of all calculations required to convert the image to raster form. Because you can not see it, the grid is called a *virtual screen.*

Any point that is part of an object descriptor, or any dimension measurement, is defined by stating its horizontal (x) and vertical (y) distances from the middle of the virtual screen (0). Each coordinate is represented by a 2-byte binary number, 15 bits of which are significant.

The maximum value that can be represented is the hexadecimal value of x'7FFE,' or the decimal equivalent of 32,766. The minimum value possible is the hexadecimal value of x'8002' or the decimal equivalent of -32,766.

Any point is located with an accuracy of one part in 32,766 (2^{15-2} locations across the entire width of the virtual screen). However, each distance is stated as a binary fraction of one, with +1 and -1 being the limits of the virtual screen as demonstrated in Figure 1-1.

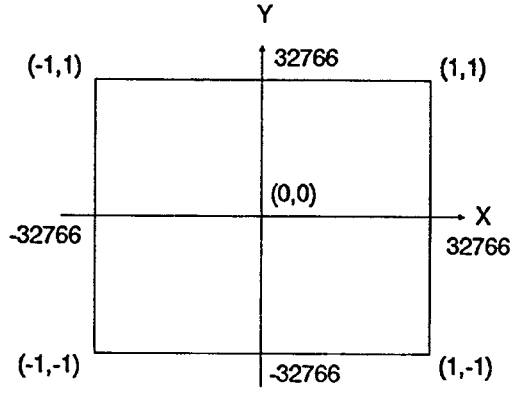


Figure 1-1 The SCODL Virtual Screen Coordinate System

Values may be positive or negative: positive x coordinates indicate points in the right half of the screen; positive y coordinates indicate points in the upper half of the screen.

Encoding Data

SCODL can receive data from disk or I/O in either binary or hexadecimal ASCII data. Binary data is not encoded as characters; it consists of ones and zeros, which are stored directly in memory.

All data received in binary mode is treated as SCODL data. Hexadecimal formats permit the use of +BGN and +END sequences which flag the beginning and end of SCODL data. These formats are useful if redundant data is to be expected before or after the actual file.

> ***NOTE:*** *Only compressed and decimal formats are supported in SCODL software releases 3.2 or earlier and are not discussed in this manual. If you require detailed information regarding these formats, refer to earlier versions of the Programmer's Guide to SCODL.*

Overview of Hexadecimal

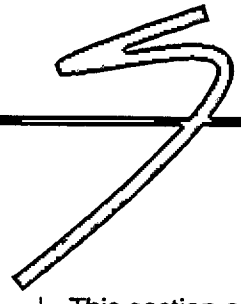
In hexadecimal format or *base 16*, the characters zero through nine and letters A through F are used as one-digit representations of the numbers 0 to 15 (16 digits altogether). This format allows each of the 16 characters to represent four bits; two characters are required to represent an eight-bit number.

Hex is a shorthand for binary notation where one hex digit represents four binary digits (bits). For example, x'FF' represents two "fifteens", or the 8 bits 1111 1111. In print, hex numbers are preceded by a lower case *x* and surrounded by single quotes to identify them.

Overview of Binary

In binary or *base 2*, computer data is notated in bit form represented by a string of zeros (0) and ones (1). These binary symbols indicate on-off states to the computer - 0 meaning OFF and 1 meaning ON, for example, 01010101.

SECTION 2
Describing Objects



This section contains:

- Describing Objects in SCODL
- Using Extended Descriptors
- Determining Coordinate Data
- Selecting Output Image
- Overlapping Priorities
- RLC Merging

Describing Objects in SCODL

SCODL formulates an image from basic geometric shapes and enables you to select appropriate characteristics such as width and color. This function is done by creating a SCODL image description that is comprised of a series of variable length object descriptors.

The descriptors indicate the following:

- Object description such as line, arc, polygon, multiple edge, and spline
- Extended description such as arc width and color
- Coordinate data indicating the location of the points that define an object in terms of the virtual screen

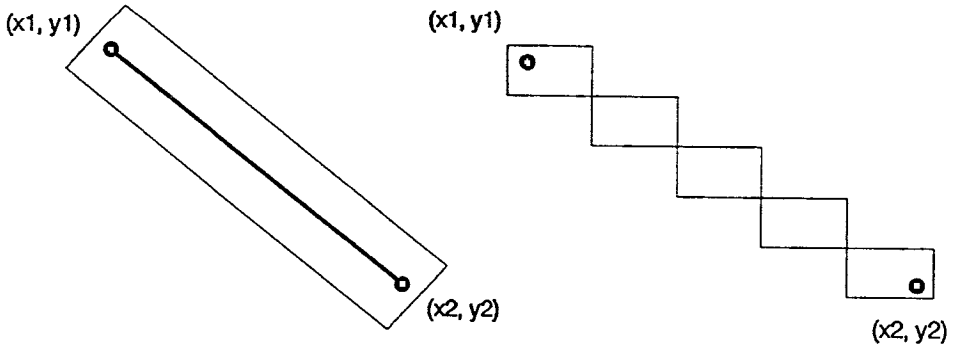
OBJECT DESCRIPTOR eg. Line, Arc, Polygon	EXTENDED DESCRIPTOR eg. arc width, color	COORDINATE DATA	END OF DESCRIPTION

Figure 2-1 Format for a SCODL Image Description

Objects Descriptors

The first byte of an image description contains opcodes that specify the shape of the object. These opcodes include:

Line (opcode 1) - Specified by two x-y coordinate end points. It appears in the final image one pixel wide at the finest resolution and in a color you select. Figure 2-2 illustrates the digital representation of a line.



(a) Theoretical Line (b) Digital representation of the Line

Figure 2-2 Digital Representation of a Line Drawn from (x1, y1) to (x2, y2)

Arc (opcodes 2,3) - Defined as the path that joins two endpoints and an intermediate point located anywhere on the arc's perimeter. The arc's chord is defined as the straight line that runs between the initial and final points. Placing the intermediate point approximately mid way between the initial and final point on the arc reduces arithmetic error in arc placement calculations.

If the three points defining the arc are co-linear, then a line is drawn between the initial and final points. If the initial and final points coincide, then a complete circle is drawn. Both the thickness and the color of the arc can be selected.

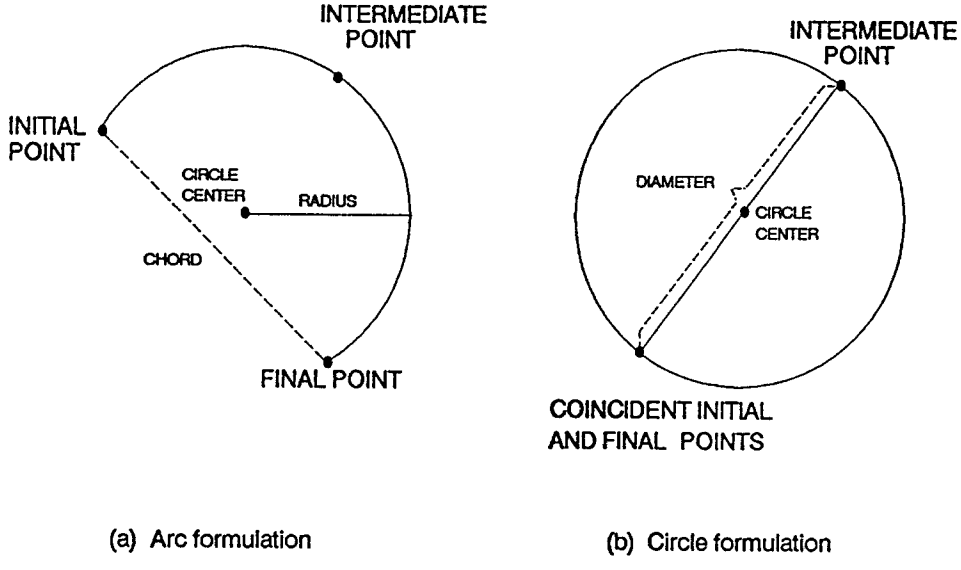
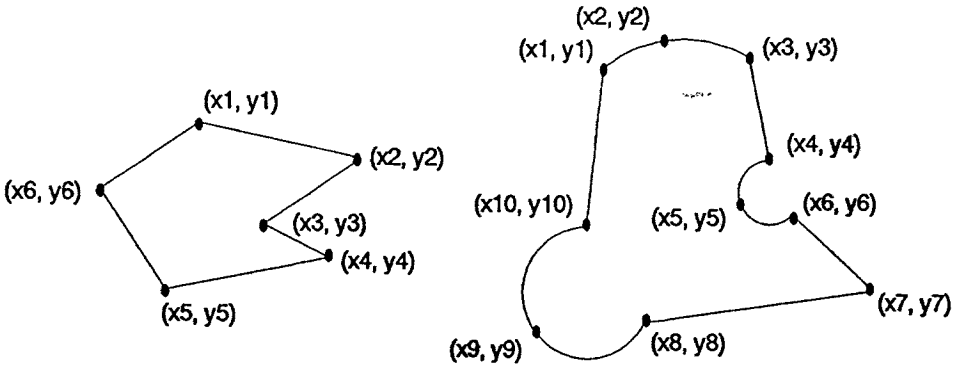


Figure 2-3 Arc and Circle Formulation

Polygon (opcodes 4, 5, 8) - Specified as either straight lines, arcs, or a mixture of both are illustrated in Figure 2-4. You can select the highlight and interior fill colors.



(a) Polygon comprised of straight lines

(b) Polygon comprised of straight lines and arc edges

Figure 2-4 Example of a Polygon Comprised of Straight Lines and Arc Edges

Two other geometric shapes are possible. These shapes are as follows:

Multiple Edge (opcode 6) - Consists of a set of connected arcs and line edges similar to a polygon, however the last coordinate pair is not automatically joined to the first to create an enclosed area.

Spline (opcodes 9, 10, 11) - A function that draws a smooth continuous curve through a series of more than three points. Splines with three points are considered regular arcs. Multiple spline edges can also be created that consist of a set of connected line edges and splines.

Using Extended Descriptors

The opcode indicating the type of object is followed by other opcodes that control the final appearance of the overall image such as color and arc width.

Color

You can specify the following two types of coloring:

- *Highlight color* - the color of the object's boundary
- *Fill color* - the color of the object's interior

The color codes used to designate highlight color and fill color are one-byte values between 0 and 255. One color code indexes into the corresponding address of four separate 256-byte look-up tables (LUTs), which give the neutral, red, green, and blue components of that color. These LUTS are necessary to adjust the red/green/blue balance of each color shade. This

adjustment ensures that each of the 256 color codes give the desired output color with the particular output system being used. The neutral LUT is used for black-and-white imaging.

The codes and the four tables constitute a palette of 256 colors, which you select out of a possible total of over 16 million color combinations. If an ink-jet printer or other types of paper output devices are used, then the LUTs only function is to serve as a color palette. A color code is translated directly into pigment intensities on the paper.

In the case of a QCR film recorder, LUTs compensate for the fact that photographic film responds with different sensitivities to the red, green, and blue components of any one color code. LUTs are in effect serving both as a color palette and as compensation tables. You can use opcode 211 (LUT compensation) and 215 (Compensation type) to compensate for these irregularities.

Default LUTs exist within the MVP+, however, you can generate LUTs to achieve perfect color results with whatever output device is being used. This process is explained in the *User's Guide to the QCR* in the section *Calculating Color Look-Up Tables*. You can load all 256 bytes of a table using opcodes 248 or 251-254 that specify neutral, red, green, or blue LUTs. LUT addresses can also be loaded by selecting opcode 225.

The contents of the LUTs can be specified at any point in the SCODL description of the image since the look-up does not take place until after the entire image description has been received.

Arc Width

After specifying color, you may use opcodes 2 and 3 to specify the arc width. This number, like a coordinate value, is a positive fraction that can only be as large as the radius of the arc. Arc widths are referred to as the highlight area of the arc, extending from the actual circle boundary (*outer boundary*) inward to the center of the circle (*inner boundary*).

The inner boundary limits how close the highlight area pixels come to the center of the arc's circle. Highlight color is applied to the entire width of the arc.

> ***NOTE:*** *If the arc width* **R-R1** *illustrated in Figure 2-5 is specified such that it is greater than the radius of the arc's circle, then the width is computed to be equal to the circle's radius size. A zero arc width (outer boundary = inner boundary) will give an arc one pixel wide.*

Every arc has, by definition, an associated *chord*. An arc's chord is a thin straight line running between the initial and final points. The chord divides the arc's circle into two parts: the arc itself and the invisible remainder of the circle.

Chords are necessary because arc imaging does not always end at the scan lines containing initial or final points. Therefore, arcs must be clipped according to the chord.

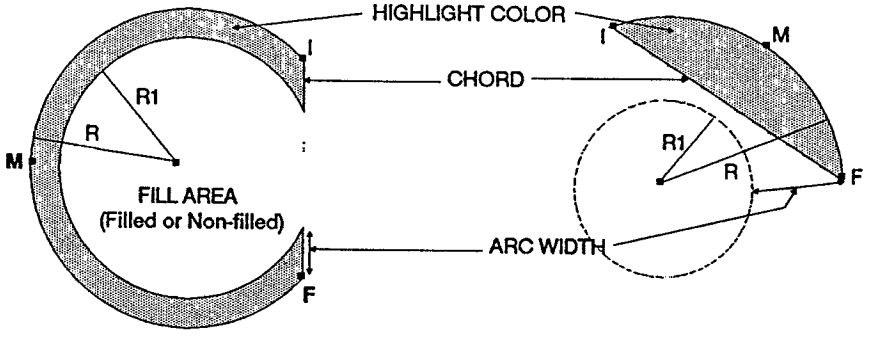


Figure 2-5 Defining Arcs

Determining Coordinate Data

Once the required object and extended descriptors have been specified, the coordinate data specifying the location of points that define the object is determined. The number of bytes of coordinate data varies with the number of points needed to define the object: a line requires two endpoints (eight bytes); arcs require three points (twelve bytes); while polygons may have various points depending on the number of sides.

The first bit of each coordinate value is reserved for a *sign bit*; **0** indicates positive, **1** indicates negative. The next fourteen bits are a binary fraction, indicating the point's distance from the virtual screen center.

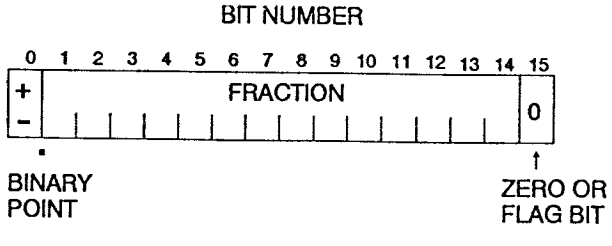


Figure 2-6 Format of Coordinate Data

Positive Coordinate Points

The remaining bits after the sign bit have successive decimal values of 2^{-1} (1/2), 2^{-2} (1/4), 2^{-3} (1/8) and so on. The maximum positive binary fraction is 0.11111111111111 (x'7FFE' hex ASCII, 32,766 decimal) located at the edge of the screen.

Negative Coordinate Points

Negative numbers on the other hand, are the 2's complement of the positive. The maximum negative binary fraction is 1.00000000000001 (0x8002 hex ASCII, -32766 decimal).

The last bit is reserved for a special purpose having to do with polygons. Set it to 0 except under conditions given in the description of the polygon opcode.

Example: Suppose a point has an X coordinate of 1/2 and a Y coordinate of -9/16. The binary point are shown here for clarity. It is not actually represented in the data, although it is assumed to be present after the first bit in any coordinate value.

In SCODL format, the binary coordinates are as follows:

```
X=1/2;      0.10000000000000
Y=-9/16;    1.01110000000000
```

How did we arrive at the -9/16 binary fraction? In SCODL format, the binary computation for this negative number is the 2's complement of +9/16 and is determined as outlined in Table 2-1.

Steps	Binary Equivalent
+9/16	0.10010000000000
1's complement	1.01101111111111
2's complement (=-9/16)	1.01110000000000

Table 2-1 Steps to Calculate 2's Complement of +9/16

Therefore, the binary fraction 1.01110000000000 represents -9/16. To verify this method, add the +9/16 binary numbers to the -9/16 numbers and check to see if it adds to zero.

Coordinate values may be subjected to further encoding, depending on the data format you selected for sending data to the MVP+. The above description is only sufficient for data sent in binary format. Refer to *Encoding Data* in Section 1 for a supplement on encoding data in hexadecimal ASCII.

The exact format for creating coordinate data for each type of geometric object is given in the descriptions of the individual opcodes found in Section 4, *Description of Opcodes*.

Selecting Output Image

After an image has been translated into SCODL (mapped onto the SCODL virtual screen), you can determine whether all of this image or only a portion is sent to the output device. The image or part of an image chosen to be sent to the output device is referred to here as the *output image.*

Opcodes 249 and 250 are used at the start of an image file to govern how much of the image on the virtual screen will be part of the output image. Both of these opcodes express values in pixels of the output device called *output pixels.* Opcode 249 requires output pixel measurements of the maximum dimensions of the virtual screen called XDIM and YDIM. Opcode 250 requires output pixel measurements of the limits of the desired output image in terms of X1, X2, Y1 and Y2.

This section explains how to derive XDIM and YDIM and then, using these values, how to derive values for X1, X2, Y1, and Y2.

Finding XDIM and YDIM (Opcode 249)

To calculate the maximum dimensions of the virtual screen, XDIM and YDIM, follow the steps below:

(a) Decide what portion of the image is to be sent to the output device and find the upper left and lower right corners of a rectangle enclosing this portion. Describe these corner points as pairs of coordinates with each coordinate expressed as a fraction of the full extent of the virtual screen according to the standard coordinate format illustrated in Figure 2-6.

Example:

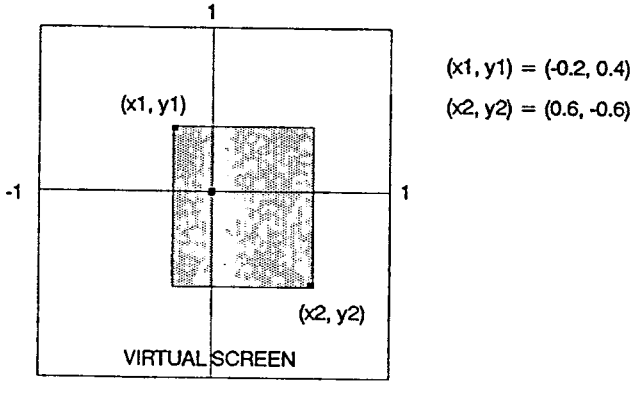


Figure 2-7 Choosing the Output Image

(b) Find the difference between the two x coordinates and the two y coordinates. This gives you two values, *a* and *b,* that represent the width and height of the output image area. They must be positive fractions less than 2.0 to remain inside the limits of the virtual screen.

$a = |x2 - x1|$ $b = |y2 - y1|$ (*absolute values*)

Example:

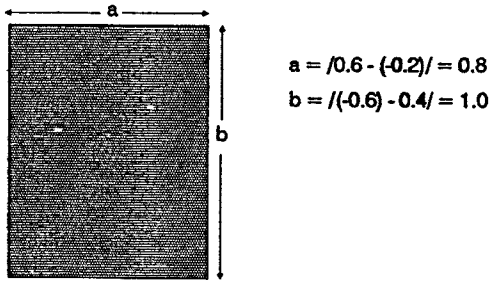


Figure 2-8 Finding Coordinate Dimensions of the Output Image

(c) Decide the number of pixels and lines the output device is expected to produce when the image is output.

Consider a Matrix Instruments QCR with a 2K resolution mode. On a frame of 4 x 5-inch film, this device has room to image a full 2048 pixels across and 1536 lines down. In either 4K mode or when using 8 x 10-inch film, these values double. For example, 35mm film has a resolution of 2048 x 1366 pixels in 2K and 4096 x 2732 pixels in 4K.

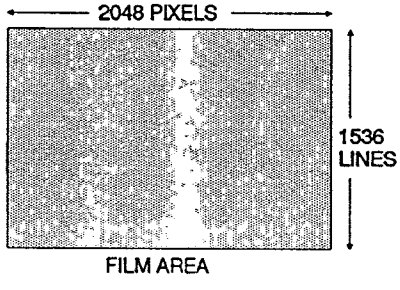


Figure 2-9 Finding Output Pixel And Line Measurements

This gives you two values: *M* and *N*. These values are the number of output pixels and lines into which the output image area you selected in (a) must be transformed. Their ratio varies depending on the film format or paper size you want to use.

Unless *M* is to *a* as *N* is to *b*, your image is elongated in one direction when it is output. This effect can be produced if desired, but results in distorted arcs. To avoid, select a smaller than maximum value for either *M* or *N*, as appropriate, such that $M/a = N/b$.

Example: The selected output image is longer than it is wide (a = 0.8, b = 1.0), while the QCR film area is wider than it is long (M = 2048, N = 1536). If you follow the calculations in (d) using these values, the selected output image is expanded in width. To avoid this distortion, assign M a value of 0.8 x 1536, or 1229.

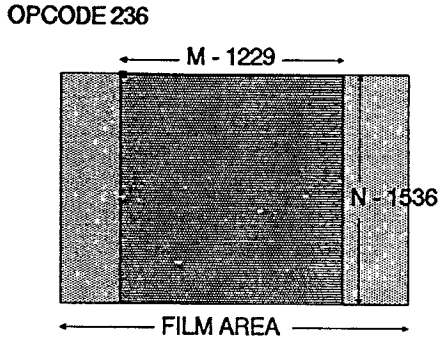


Figure 2-10 Fitting the Output Image Into The Film Or Paper Area

The output image does not fill the entire film area. If you want it centered on the film, use opcode 236 (Device Dependent) to specify an appropriate start pixel. This also works for ink-jet printers.

Opcode 226 (Virtual Screen Image Rotation) allows objects in the image to be rotated around the virtual screen center. Rotating by 90 or 270 degrees turns the virtual screen image onto its right or left side; this may be used to help fit a long, tall image into a short, fat output area or vice versa.

(d) Using the following equations, find XDIM and YDIM. Given the decisions you have made so far, these values represent the number of output pixels needed to image the entire virtual screen. If your results are not whole numbers, round them down.

Example:

XDIM = 2M/a XDIM = 2(1229)/0.8 = 3072
YDIM = 2N/b YDIM = 2(1536)/1.0 = 3072

These values may now be used in a subcommand of opcode 249 at the start of an image file. The defaults for these values in case you do not specify any are 2048 in 2K mode and 4096 in 4K mode.

> ***NOTE:*** *If you decided in step (c) to avoid image distortion by making M/a = N/b, XDIM and YDIM should be equal as in the example above.*

Finding X1, X2, Y1, Y2 (Opcode 250)

To calculate the limits of the desired output image in terms of X1, X2, Y1, and Y2 follow the steps below:

(e) Return to the two coordinate pairs, which you selected in (a). These points defined the part of the virtual screen, which you selected to be output. Use the equations below to change these points from the SCODL coordinate format into numbers of output pixels. The results will be called X1, Y1, X2, Y2. Again, if your results are not whole numbers, round them down.

Example:

X1 = x1(XDIM/2)	X1 = -0.2 (1536) = -307
X2 = x2(XDIM/2)	X2 = 0.6 (1536) = 921
Y1 = y1(YDIM/2)	Y1 = 0.4 (1536) = 614
Y2 = y2(YDIM/2)	Y2 = -0.6 (1536) = -921

(f) The calculations can be verified through the following equations:

Example:

X2 - X1 + 1 = M	921 - (-307) + 1 = 1229
Y1 - Y2 + 1 = N	614 - (-921) + 1 = 1536

These verification equations must balance. It is possible that they will balance with the first values used; if not, adjust X1, X2, Y1, and Y2 by a value of one or two until they balance. The values may now be used in opcode 250 (Output Image Dimensions) at the start of an image file. The 2K mode defaults for these values are as follows:

X1 = -1024 = x'FC00'
X2 = 1023 = x'03FF'
Y1 = 767 = x'02FF'
Y2 = -768 = x'FD00'

Overlapping Priorities

As each object descriptor is received, it generates an object that is drawn on the screen. The priority of an object is determined by its temporal position in the data stream of received object descriptors. Object descriptors may partially or completely overwrite previously defined objects. The last object drawn will always be completely visible.

Figure 2-11 displays only a triangle.

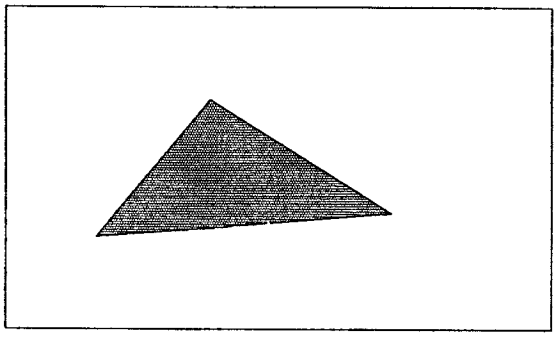


Figure 2-11

Figure 2-12 displays part of the triangle being obscured by the circle.

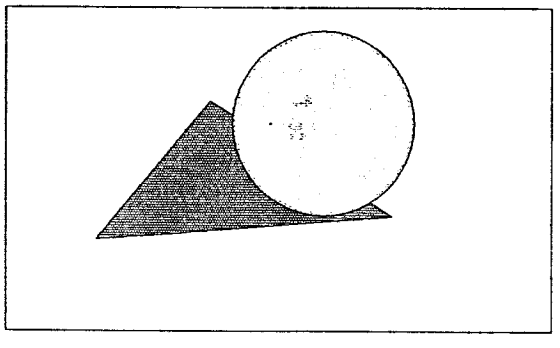


Figure 2-12

Figure 2-13 demonstrates a hole (transparent) polygon, which allows previously drawn objects to show through.

Refer to the description of opcode 5 in Section 4, *Description of Opcodes* for more information on this exception to the overlap rule.

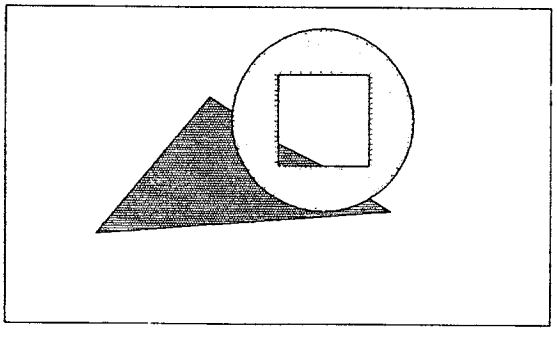


Figure 2-13

Figures 2-11 to 2-13 Priorities of Overlapping Objects

RLC Merging

Pixel Mode versus RLC Data

In pixel mode, images are described pixel by pixel; each pixel is represented by a separate intensity (color) byte. However, Run Length Codes (RLCs) allow each line of an image to be described as a series of segments of a given color. Each segment is described by two bytes. The first byte specifies the segment length from 1 to 255 pixels, and the second byte specifies the segment intensity from 0 to 255.

Using this method, RLC data is more efficient at handling images containing many uniform colors than in pixel mode. Less memory space is required to store the image.

Technique and Use of RLC Merging

If all RLCs of one color are made transparent, every matching color code in the image will also become transparent. SCODL allows the user to place such an RLC image on top of an existing SCODL image. This creates an overlay effect in which parts of one image show through parts of an overlying image.

You must specify the desired area of the RLC image to be overlaid, the transparent color in the RLC image. You are then ready to send the RLC image data. This procedure can be prepared in advance by creating a SCODL file using either the MVP+ or by other means.

In *Overlapping Priorities* we saw that SCODL allows hole polygons to be specified, creating transparent areas in existing SCODL objects. The advantage of RLC merging over hole polygons is that it allows for overlaying of very complex shapes and images that could not easily be built out of successive solid and transparent polygons. Two limitations of RLC merging include: overlaid images must be prepared separately from the underlying image; and that only one color in the overlaid image can be made transparent.

To begin, you can create a box of RLC data to be overlaid onto an image. This is done by *spooling* an output image onto disk from the MVP+. The file stored on disk can be used as RLC merging box data, as described below.

> ***NOTE:*** *Only two-byte RLCs each RLC having one pixel count byte such as those used by the Matrix Instruments QCR can be used for RLC merging.*

The procedure described on the following page represents one way of using RLC merging by alternating RLC boxes with SCODL objects. However, it is not necessary to have SCODL data other than the background color polygon before the first RLC box. It is also not necessary to place any SCODL data between RLC boxes. The RLC box does not need to include any part of the selected output image area.

Merging an RLC box of a given resolution with an image of a higher resolution causes the RLC box to appear as a proportionally smaller part of the final image.

1. Create a SCODL image.
   Note: Do not send end-of-image code (128) yet.
2. Select transparent color (221). Default is 0.
3. Specify RLC image boundaries (220). Default is current output image area.
4. Input the RLC data (219). SCODL image shows through all (221) areas of transparent color code.
5. If desired, add another SCODL image. Remember that the background color of any SCODL image following RLC merging is transparent.
6. If desired, add another RLC box (219). The transparent color (221) and RLC image boundaries (220) will remain the same unless respecified.
7. Add additional SCODL images and RLC boxes alternately as required.
8. When image is complete, send (128) - x'8000' to begin MVP+ processing.

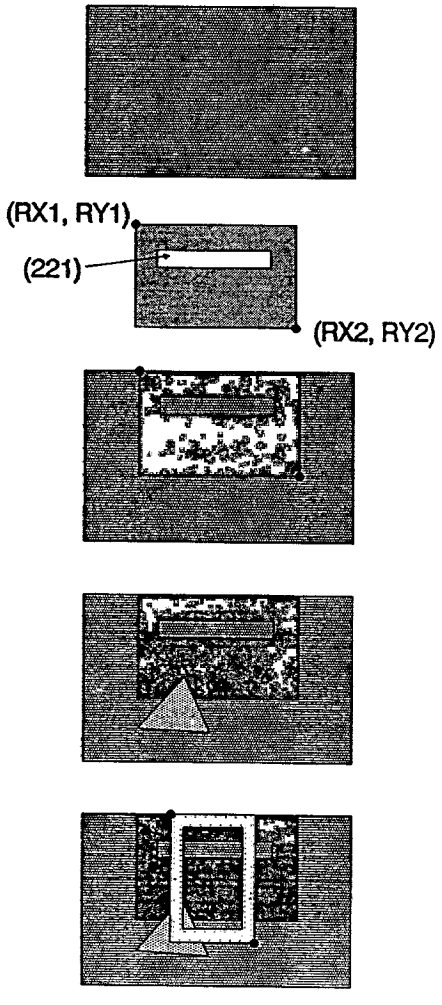
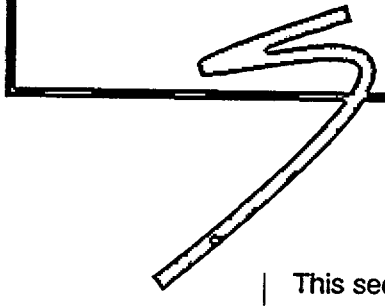


Figure 2-14 Procedure for Using R LC Merging

SECTION 3
Describing Macros and Text



This section contains:

- About Macros and Text
- Macro Commands
- Defining Macros and Typefaces
- Using the Default Character Font

About Macros and Text

Macros are packages of sequential SCODL commands that can be summoned by a single SCODL command called a *macro request*. Macros are usually one or more object descriptors, extended descriptors, and coordinate data that are applicable to many images and have therefore been stored permanently within SCODL.

Macros save time and reduce the complexity of larger SCODL descriptions. Macros are primarily used for, but are not limited to, describing character sets. The commands included in SCODL for manipulating macros are primarily designed to allow you to include text in images.

Macros are identified by two one-byte codes. The family/font code classes the macro as one of a particular family. For text, this family (0 to 127) generally includes all the macros of a specific style or *font*. Macro codes (1 to 254) can describe alphabetical characters, punctuation marks, or mathematical symbols.

SCODL contains a set of macros that makes up font 0. These sets are included in SCODL enabling you to easily add text to images without specifying each character as a set of object descriptors. Appendix A, *Default Character Font* lists all the characters currently contained in this font.

This section introduces the opcodes that control the use of macros. Each opcode is explained in detail in Section 4, *Description of Opcodes.*

Macro Commands

Table 3-1 lists the macro definition commands used to create macros. Similarly, Table 3-2 lists the possible commands that can be used in a macro request.

Dec	Hex	Command
214	D6	Specify a kerning table
217	D9	Font attributes
227	E3	Define a macro
241	F1	End of macro definition
247	F7	Set kerning points

Table 3-1 Macro Definition Commands

Dec	Hex	Command
130	82	Macro string length
206	CE	Macro access mode
210	D2	Macro color control
222	DE	Macro string width
230	E6	Catenate string
231	E7	Select font
232	E8	Macro rotation
233	E9	String spacing
234	EA	Begin a string
237	ED	Delete macro font
240	F0	Macro request
242	F2	Set macro colors
243	F3	Macro base address
244	F4	Macro scale

Table 3-2 Macro Request, Location, Size, and Color Commands

Suppose, in the course of entering SCODL data describing an image, you want to add one or more of the characters in the default typeface listed in Appendix A to the image. Among the features you must specify for these characters are size, location in the image, and color. The commands that specify these features are as follows:

210	D2	Macro color control
244	F4	Macro scale
243	F3	Macro base address
242	F2	Set macro colors

CAUTION: *Do not request a macro or a macro string in a size and location that places part of a macro beyond the edge of the virtual screen. The results are undefined.*

Unless otherwise specified, macros are positioned right-side-up. However, all macros requested can be rotated by a given angle of rotation until a different rotation command is specified by using the following command:

232	E8	Macro rotation

The following command is available to thicken lines and arcs within macros, giving them a bolder appearance. Be sure to observe the precautions given in the description of this opcode before using it.

216	D8	Thicken Edge

Single Macros or Characters

Once these features have been specified, macros can be requested one at a time with the following command:

240	F0	Macro request

Strings or Lines of Text

Since most uses of characters involve words or lines of text, additional commands exist to let you request and manipulate a number of macros in a string. To request a string, you must first specify its size, color, location and, if desired, rotation. You must then select a string mode and spacing arrangement. This is done with the following opcode:

233	E9	String spacing

You may have a specific width within which you want a string to fit, or you may want a number of strings to align on one or both edges. An opcode is available for these cases:

222	DE	Macro string width

Instead of requesting the string of macros one at a time using opcode 240, you can use the following commands:

231	E7	Select font
234	EA	Begin a string

The font requested by opcode 231 as well as the scale, base address, and colors that were previously in effect is applied to all macros in the string.

If you finish the string command by sending the x'FF' marker to signify the end of opcode 234 and you want to add more characters to the string, the following opcode is available:

230	E6	Catenate string

This opcode allows another string to be tagged on behind the last one without respecifying the spacing mode, font, or base address. However, any previously specified left, right, or center justification is ignored.

Defining Macros and Typefaces

A macro packages a frequently used series of SCODL objects and/or commands into two bytes: a font/family code and a macro code. You can program SCODL commands into any macro code from 1 to 254 and in any family from 1 to 127. Remember that the family of internal characters with family/font code 0 can not be redefined.

If the macro to be defined is going to be a member of a typeface, the whole family should be designed so that their vertical placement in a string is typographically correct. The following command sets base and cap lines for the whole family/font:

217	D9	Specify Font Attributes

If the macro to be defined is going to be used in kerned, proportionally spaced strings, it is necessary to define the edges of the macro. Defining the edges is important since this is where the inter character spacing is measured from.

247	F7	Set kerning points
214	D6	Specify a kerning table

Now you can define a macro using this command:

227	E3	Define a macro

This command gives the family/font and macro codes, followed by the SCODL data making up the macro. Note the restrictions on the contents of macro definitions given in the description of this opcode. The SCODL data making up the macro or character is followed by:

241	F1	End of macro definition

Macro storage requires memory space that is taken from general system memory. Storage of a large number of macros may affect the performance of the system and, therefore, the number that can be stored at once is ultimately limited. Defined macros may be erased by using a (227) Define Macro command containing no data followed by the End of Macro Definition code. Fonts can be deleted from memory, **not** from disk by using:

237	ED	Delete macro family/font

Font Choices

SCODL comes equipped with its own specified internal character font. The font macro descriptions and kerning sizes are provided in *Default Character Font* explained later in this section. Eight additional fonts are supplied for the MVP+ on a separate diskette from the MVP+ software.

The fonts included with the MVP+ software are listed in Table 3-3:

Font #	Type
1	Times
2	Times Bold
3	Triumvirate
4	Triumvirate Bold
5	Swiss - sans serif
7	Swiss Bold - sans serif
17	Dutch Roman - serif
19	Dutch Bold - serif

Table 3-3 MVP+ Supplied Fonts

These fonts and their kerning information are described in the *MVP+ User's Guide.*

Using the Default Character Font

Table 3-4 provides the sizes and kerning points for the SCODL internal character set. For each character, the eight kerning point values are given in the order used by opcode 247. The first two numbers are the distances that the character extends above and below the virtual screen center; the next three numbers are the three left kerning points; and the last three numbers are the three right kerning points.

The numbers are expressed as positive fractions of the distance from the center to the edges of the virtual screen on which the macros are defined. The left and right kerning points of the internal macro characters have been determined according to a formula.

The top pair of kerning points are taken from the portions of the macro's left and right edges, which are above the line defined by *y=0.45* in the macro definition. The top left and top right kerning points are the leftmost and rightmost extremes, respectively, of the macro's edges within this region.

The middle pair of kerning points are taken from the portions of the macro's edges that are between the lines defined by *y=0.45* and *y=-0.18* in the macro definition. The middle kerning points are the leftmost and rightmost extremes of the macro's edges within this region. The bottom pair of kerning points are taken from the region below the line defined by *y=-0.18* in the macro definition.

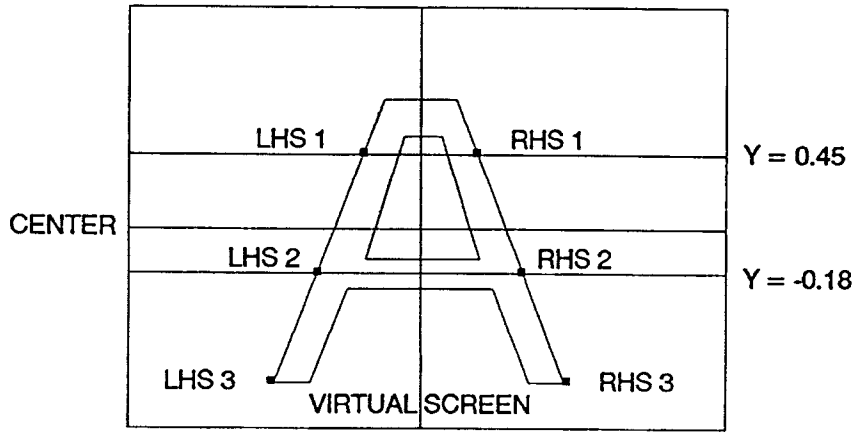


Figure 3-1 Kerning Points of the Internal Font

Macro Code (Hex)	(Dec)		Width	Kerning Points (top, Bottom, 3 Left, 3 Right)	Name
0018	24	.	0.44	0.35 0.08 0.22 0.22 0.22 0.22 0.22 0.22	bullet
0019	25	"	0.68	0.79 0.18 0.34 0.34 0.34 0.34 0.34 0.34	neut dbl quote
001A	26	'	0.29	0.79 0.18 0.15 0.15 0.15 0.14 0.14 0.14	neut sgl quote
001B	27	"	0.68	0.79 0.18 0.34 0.34 0.34 0.34 0.34 0.34	left dbl quote
001C	28	←	1.40	0.82 0.57 0.70 0.70 0.70 0.70 0.70 0.70	left arrow
001D	29	→	1.40	0.82 0.57 0.70 0.70 0.70 0.70 0.70 0.70	right arrow
001E	30	↑	1.40	0.82 0.57 0.70 0.70 0.70 0.70 0.70 0.70	up arrow
001F	31	↓	1.40	0.82 0.57 0.70 0.70 0.70 0.70 0.70 0.70	down arrow
0020	32		0.60	0.78 0.58 0.30 0.30 0.30 0.30 0.30 0.30	blank space
0021	33	!	0.38	0.82 0.55 0.19 0.19 0.19 0.19 0.19 0.19	exclamation mark
0022	34	"	0.68	0.79 0.18 0.34 0.34 0.34 0.34 0.34 0.34	right dbl quote
0023	35	#	0.72	0.57 0.34 0.36 0.36 0.36 0.36 0.36 0.36	hash mark
0024	36	$	1.20	0.98 0.72 0.60 0.60 0.60 0.60 0.60 0.60	dollar sign
0025	37	%	0.94	0.82 0.55 0.55 0.55 0.55 0.55 0.39 0.55	percent sign
0026	38	&	1.22	0.84 0.58 0.42 0.60 0.61 0.30 0.53 0.61	ampersand
0027	39	'	0.30	0.79 0.18 0.15 0.15 0.15 0.15 0.15 0.15	apostrophe
0028	40	(	0.54	0.98 0.71 0.20 0.27 0.25 0.27 0.08 0.27	left bracket
0029	41	)	0.54	0.98 0.71 0.27 0.09 0.18 0.20 0.27 0.25	right bracket
002A	42	*	0.77	0.50 0.27 0.39 0.39 0.39 0.38 0.38 0.38	asterisk
002B	43	+	0.72	0.48 0.25 0.36 0.36 0.36 0.36 0.36 0.36	plus sign
002C	44	,	0.30	0.00 0.88 0.15 0.15 0.15 0.15 0.15 0.15	comma
002D	45	-	0.72	0.21 0.03 0.36 0.36 0.36 0.36 0.36 0.36	minus sign
002E	46	.	0.25	0.00 0.55 0.13 0.13 0.13 0.12 0.12 0.12	period
002F	47	/	0.72	0.82 0.55 0.00 0.22 0.36 0.36 0.27 0.06	divide sign
0030	48	0	1.20	0.82 0.58 0.60 0.60 0.60 0.60 0.60 0.60	zero
0031	49	1	1.20	0.78 0.55 0.60 0.60 0.60 0.60 0.60 0.60	one
0032	50	2	1.20	0.83 0.55 0.60 0.60 0.60 0.60 0.60 0.60	two
0033	51	3	1.20	0.82 0.58 0.60 0.60 0.60 0.60 0.60 0.60	three
0034	52	4	1.20	0.81 0.55 0.60 0.60 0.60 0.60 0.60 0.60	four
0035	53	5	1.20	0.78 0.58 0.60 0.60 0.60 0.60 0.60 0.60	five
0036	54	6	1.20	0.83 0.59 0.60 0.60 0.60 0.60 0.60 0.60	six
0037	55	7	1.20	0.79 0.55 0.60 0.60 0.60 0.60 0.60 0.60	seven
0038	56	8	1.20	0.81 0.58 0.60 0.60 0.60 0.60 0.60 0.60	eight
0039	57	9	1.20	0.83 0.59 0.60 0.60 0.60 0.60 0.60 0.60	nine
003A	58	:	0.30	0.15 0.55 0.15 0.15 0.15 0.15 0.15 0.15	colon
003B	59	;	0.30	0.15 0.88 0.15 0.15 0.15 0.15 0.15 0.15	semi-colon
003C	60	<	0.87	0.81 0.57 0.44 0.44 0.44 0.43 0.43 0.43	less than sign
003D	61	=	0.72	0.35 0.11 0.36 0.36 0.36 0.36 0.36 0.36	equal sign
003E	62	>	0.86	0.81 0.57 0.43 0.43 0.43 0.43 0.43 0.43	greater than sign

Table 3-4 List of SCODL Internal Macros (continued...)

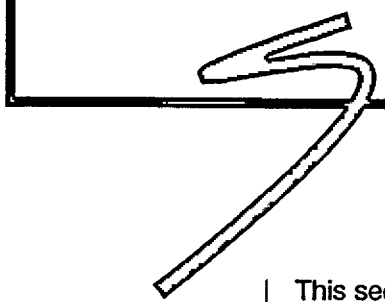
(Hex) (Dec) (Continued...)			Width	Kerning Points (top, Bottom, 3 Left, 3 Right)	Name
003F	63	?	1.00	0.82 0.55 0.36 0.50 0.11 0.47 0.50 0.15	question mark
0040	64	¢	0.97	0.80 0.54 0.49 0.49 0.49 0.48 0.48 0.48	cent sign
0041	65	A	1.24	0.82 0.55 0.25 0.47 0.62 0.24 0.47 0.62	upper case A
0042	66	B	1.10	0.82 0.55 0.55 0.55 0.55 0.50 0.55 0.55	upper case B
0043	67	C	1.24	0.86 0.59 0.53 0.62 0.59 0.57 0.60 0.62	upper case C
0044	68	D	1.16	0.82 0.55 0.58 0.58 0.58 0.48 0.58 0.55	upper case D
0045	69	E	1.02	0.82 0.55 0.51 0.51 0.51 0.50 0.41 0.51	upper case E
0046	70	F	0.96	0.82 0.55 0.48 0.48 0.48 0.48 0.39 0.21	upper case F
0047	71	G	1.28	0.86 0.59 0.53 0.64 0.61 0.57 0.64 0.64	upper case G
0048	72	H	1.10	0.82 0.55 0.55 0.55 0.55 0.55 0.55 0.55	upper case H
0049	73	I	0.28	0.82 0.55 0.14 0.14 0.14 0.14 0.14 0.14	upper case I
004A	74	J	0.88	0.82 0.58 0.00 0.44 0.44 0.44 0.44 0.44	upper case J
004B	75	K	1.20	0.82 0.55 0.60 0.60 0.60 0.54 0.29 0.60	upper case K
004C	76	L	1.00	0.82 0.55 0.50 0.50 0.50 0.00 0.00 0.50	upper case L
004D	77	M	1.36	0.82 0.55 0.68 0.68 0.68 0.68 0.68 0.68	upper case M
004E	78	N	1.10	0.82 0.55 0.55 0.55 0.55 0.55 0.55 0.55	upper case N
004F	79	O	1.32	0.86 0.59 0.56 0.66 0.63 0.55 0.66 0.63	upper case O
0050	80	P	1.04	0.82 0.55 0.52 0.52 0.52 0.47 0.52 0.25	upper case P
0051	81	Q	1.32	0.86 0.63 0.56 0.66 0.64 0.54 0.65 0.66	upper case Q
0052	82	R	1.08	0.82 0.55 0.54 0.54 0.54 0.51 0.53 0.54	upper case R
0053	83	S	1.16	0.87 0.59 0.51 0.58 0.58 0.54 0.58 0.58	upper case S
0054	84	T	1.10	0.82 0.55 0.55 0.14 0.14 0.55 0.14 0.14	upper case T
0055	85	U	1.10	0.82 0.59 0.55 0.55 0.55 0.55 0.55 0.55	upper case U
0056	86	V	1.20	0.82 0.55 0.60 0.50 0.28 0.60 0.51 0.30	upper case V
0057	87	W	1.74	0.82 0.55 0.87 0.79 0.60 0.87 0.80 0.61	upper case W
0058	88	X	1.20	0.82 0.55 0.57 0.40 0.60 0.55 0.39 0.60	upper case X
0059	89	Y	1.20	0.82 0.55 0.60 0.45 0.13 0.60 0.46 0.16	upper case Y
005A	90	Z	1.14	0.82 0.55 0.55 0.39 0.57 0.53 0.51 0.57	upper case Z
005B	91	[	0.50	0.98 0.71 0.25 0.25 0.25 0.25 0.25 0.25	left sq bracket
005C	92	\	0.72	0.82 0.55 0.36 0.27 0.06 0.00 0.21 0.36	backslash
005D	93	]	0.50	0.98 0.71 0.25 0.25 0.25 0.25 0.25 0.25	right sq bracket
0060	96	'	0.29	0.79 0.18 0.15 0.15 0.15 0.14 0.14 0.14	back apostrophe
0061	97	a	0.96	0.46 0.58 0.01 0.43 0.48 0.01 0.43 0.48	lower case a
0062	98	b	0.98	0.82 0.58 0.49 0.49 0.49 0.00 0.49 0.49	lower case b
0063	99	c	0.96	0.49 0.58 0.01 0.48 0.48 0.01 0.48 0.48	lower case c
0064	100	d	0.98	0.82 0.58 0.00 0.49 0.49 0.49 0.49 0.49	lower case d
0065	101	e	1.00	0.48 0.58 0.00 0.50 0.50 0.00 0.50 0.50	lower case e
0066	102	f	0.60	0.82 0.55 0.16 0.30 0.16 0.30 0.30 0.11	lower case f
0067	103	g	1.00	0.40 0.99 0.00 0.50 0.50 0.00 0.50 0.50	lower case g

Table 3-4 List of SCODL Internal Macros (continued...)

(Hex)	(Dec)		Width	Kerning Points (top, Bottom, 3 Left, 3 Right)	Name
(Continued...)					
0068	104	h	0.90	0.82 0.55 0.45 0.45 0.45 0.00 0.45 0.45	lower case h
0069	105	i	0.27	0.82 0.55 0.14 0.14 0.14 0.14 0.14 0.14	lower case i
006A	106	j	0.44	0.82 0.88 0.05 0.05 0.22 0.22 0.22 0.22	lower case j
006B	107	k	0.96	0.82 0.55 0.48 0.48 0.48 0.00 0.45 0.48	lower case k
006C	108	l	0.28	0.82 0.55 0.14 0.14 0.14 0.14 0.14 0.14	lower case l
006D	109	m	1.44	0.47 0.55 0.00 0.72 0.72 0.00 0.72 0.72	lower case m
006E	110	n	0.88	0.47 0.55 0.00 0.44 0.44 0.00 0.44 0.44	lower case n
006F	111	o	1.02	0.48 0.58 0.00 0.51 0.51 0.00 0.51 0.51	lower case o
0070	112	p	0.98	0.48 0.87 0.00 0.49 0.49 0.00 0.49 0.49	lower case p
0071	113	q	0.98	0.48 0.87 0.00 0.49 0.49 0.00 0.49 0.49	lower case q
0072	114	r	0.58	0.48 0.55 0.00 0.29 0.29 0.00 0.29 0.00	lower case r
0073	115	s	0.90	0.50 0.58 0.00 0.43 0.45 0.00 0.42 0.45	lower case s
0074	116	t	0.58	0.75 0.56 0.15 0.29 0.29 0.11 0.29 0.29	lower case t
0075	117	u	0.88	0.44 0.58 0.00 0.44 0.44 0.00 0.44 0.44	lower case u
0076	118	v	0.92	0.44 0.55 0.00 0.46 0.28 0.00 0.46 0.29	lower case v
0077	119	w	1.42	0.44 0.55 0.00 0.71 0.56 0.00 0.71 0.56	lower case w
0078	120	x	0.99	0.44 0.55 0.00 0.45 0.49 0.00 0.46 0.50	lower case x
0079	121	y	0.96	0.44 0.88 0.00 0.48 0.29 0.00 0.48 0.30	lower case y
007A	122	z	0.86	0.45 0.55 0.00 0.36 0.43 0.00 0.41 0.43	lower case z

Table 3-4 List of SCODL Internal Macros

SECTION 4
Description of Opcodes



This section contains:

- Opcode Rules
- Startup and End of Image Defaults
- List of Opcodes by Opcode Number

Plus:

- Opcode descriptions for each individual opcode arranged in ascending decimal order by their opcode decimal number.

Opcode Rules

The following subsections specify the SCODL opcode priorities and the language format for each of the opcodes.

Opcode Priorities

Tables 4-1 to 4-3 below places the opcodes into three priority groups - Group **A**, **B**, and **C**. Read the following and make sure you follow these priorities.

Group A opcodes: These opcodes must appear before any Group B opcodes.

Dec	Hex	Meaning
211	D3	LUT compensation control
212	D4	RLC mapping data
213	D5	RLC mapping parameters
214	D6	Kerning table
217	D9	Font attributes
218	DA	Background color
223	DF	Select output device
224	E0	SCODL control
226	E2	Set image rotation
227	E3	Define a macro
237	ED	Delete macro font
238	EE	Set frame number
239	EF	Set SCODL tables
241	F1	End of macro definition
245	F5	Image repeat
246	F6	GPIB pass through
247	F7	Set kerning points
249	F9	Virtual screen maximum dimensions
250	FA	Output image dimensions

Table 4-1 Group A Opcode Priorities

Group B opcodes: These opcodes must appear after any Group A:

Dec	Hex	Meaning
1	01	Single edge (line)
2	02	Arc
3	03	Filled arc
4	04	Polygon
5	05	Hole polygon
6	06	Multiple edge
7	07	Flood
8	08	Compressed polygon
9	09	Polygon with splines
10	A	Hole polygon with splines
11	B	Multiple spline edge
130	82	Return macro string length
206	CE	Macro access code
210	D2	Macro color control
219	DB	RLC merging data
220	DC	RLC box max dimensions
221	DD	RLC box transparent color
222	DE	Macro string width
230	E6	Catenate string
231	E7	Select font
232	E8	Macro rotation
233	E9	String spacing
234	EA	Begin a string
235	EB	Weighting function
240	F0	Macro request
242	F2	Set macro colors
243	F3	Macro base address
244	F4	Macro scale

Table 4-2 Group B Opcode Priorities

Group C opcodes: These opcodes may appear anywhere in the image file:

Dec	Hex	Meaning
0	00	Null command
128	80	End of image code
129	81	SCODL status report request
215	D7	LUT compensation type
216	D8	Thicken edge
225	E1	Load sel LUT values
236	EC	Device dependent
248	F8	Load three LUTs
251-4	FB-FE	Load LUTs individually
255	FF	Null command

Table 4-3 Group C Opcode Priorities

Startup and End of Image Defaults

Table 4-4 lists opcodes that are reset upon MVP+ startup. Table 4-5 lists opcodes that may be placed between images. At power up or at the start of a new image, these opcodes must be specified if the default is not desired. All of these opcodes may be replaced by a subsequent opcode of this type.

Dec	Hex	Opcode	Default
211	D3	LUT comp control	2 - no compensation
215	D7	Compensation type	Linear LUT used - no actual comp.
216	D8	Thicken edge	0 - no thickening
217	D9	Font attributes	Xbase/Xcap to bottom/top of screen
218	DA	Background color	Color code 0
220	DC	RLC box dimensions	Same as current output image dimensions
221	DD	RLC box transp color	Color code 0
224	E0	SCODL control	User's macros erased; SCODL table sizes reset to 8000/100/500/100/1024
226	E2	Set image rotation	0 degrees
236	EC	Device dependent	See opcode description
238	EE	Set frame number	Resets to 1
239	EF	Set SCODL tables	Reset to 8000/100/500/100/1024
245	F5	Image repeat	Resets to 1 after all repeats of image, or if the image is aborted, or at start-up.
249	F9	Virtual scrn max dimens	2K: x'0800' 4K: x'1000'
250	FA	Output image dimens	See opcode description
225, 248,251-254		LUT commands	Linear LUTs

Table 4-4 Features Reset on MVP+ Startup

Dec	Hex	Opcode	Default
210	D2	Macro color control	2 - colors specified by opcode (242)
231	E7	Select font	0 - the internal character set
232	E8	Macro rotation	0 - no rotation
233	E9	String spacing	0 - no inter-character spacing
235	EB	Weighting function	0 - polygon will be drawn as defined
242	F2	Set macro colors	Color code 0 for both highlight and fill
243	F3	Macro base address	0 - virtual screen center
244	F4	Macro scale	0 - no scale

Table 4-5 Features Reset at End of Image

List of Opcodes, by Opcode Number

Table 4-6 lists all opcodes in ascending order by their decimal value. Each opcode is listed with its hexadecimal equivalent, the number of required bytes and a brief statement of the opcode contents.

> ***HINT:*** *For quick reference, use Table 4-6 or Appendix D to locate desired opcodes.*

Dec	Hex	Opcode	Bytes	Contents
0	00	Null command	1	---
1	01	Single edge (line)	10	Color, 2 coord. pts.
2	02	Non-filled arc	16	Color, width, 3 coord. pts.
3	03	Filled arc	18	Highlight, 0 byte, fill, width, 3 coord. pts.
4	04	Polygon	n	Highlt, 0 byte, fill, coord. pts.
5	05	Hole polygon	n	Coord. pts.
6	06	Multiple edge	n	Color, coord. pts.
7	07	Flood	6	Flood color, seed pt. X, Y coord. pts.
8	08	Compressed mode polygon	n	One-byte differential coord. pts.
9	09	Polygon with spline curves	n	Color, coord. pts
10	A	Hole polygon with spline curves	n	Coord. pts
11	B	Multiple spline edge	n	Color, coord. pts
128	80	End of image description marker	2	Zero byte
129	81	SCODL status report request	2	Subcommand to turn on/off SCODL status
130	82	Return macro string length	n	Mode bits
206	CE	Macro access mode	4	Access mode, font selection code
210	D2	Macro color control	2	1=internal colors; 2=specified colors

Table 4-6 List of SCODL Opcodes by Opcode Number (continued...)

Dec (continued...)	Hex	Opcode	Bytes	Contents
211	D3	LUT compensation control	2	1=on; 2=no compensation
212	D4	RLC mapping data	n	N bytes of RLC image data
213	D5	RLC mapping parameters	14	Coord. pts, max pixels and scan line values
214	D6	Specify kerning table	n	Font code, zero byte, N, 4-byte entries
215	D7	Select type of LUT compensation	n	Subcommand byte, code or option LUT
216	D8	Thicken edge	4	Subcommand, 2-byte thickening value
217	D9	Specify font attributes	7	Family/font, 0, X-base val, X-cap value
218	DA	Specify background color	3	Zero byte, color code (0)
219	DB	RLC merging box data	n	Zero byte, RLC data, x'00FF'
220	DC	RLC merging box dimensions	10	Zero byte, 2 X coords., 2 Y coordinates
221	DD	Transparent color of RLC box	3	Zero byte, color code
222	DE	Macro string width	3	String width
223	DF	Select output device	3	Output device code, resolution
224	E0	SCODL control command	2	Subcommand byte
225	E1	Load selected LUT values	n	Target LUT, 1st location, # bytes, new values
226	E2	Set virtual screen image rotation	2	Rotation mode 0 to 3 (0(-270()
227	E3	Define a macro	3	Family/font, macro code
230	E6	Catenate macro string	n	Macro requests, x'FF'
231	E7	Select string font	2	Family/font (default=0)
232	E8	Macro rotation	5	Cos and sin of rotation angle (0()
233	E9	Specify string mode and spacing	5	Spacing mode, text alignment, spacing
234	EA	Begin a macro string	n	Macro requests, x'FF'
235	EB	Compressed mode weighting fcn	3	Zero byte, weighting function (0w8)
236	EC	Output device dependent	6	Start pixel X & Y coord. pts.
237	ED	Delete macro family/font	3	Zero byte, family/font code
238	EE	Specify frame reference number	3	2-byte frame reference number
239	EF	Specify SCODL table sizes	11	5 values (8000/100/500/100/1024)
240	F0	Macro request	3	Family/font, macro code
241	F1	End of macro definition	1	---
242	F2	Specify macro colors	3	Highlight color, fill color
243	F3	Specify macro base address	5	X, Y coords of base address (0)
244	F4	Specify macro scale	5	X, Y scale factors for macro coordinates
245	F5	Image repeat request	2	Repeat count of 1-255
246	F6	GPIB data pass through	n	Address, read, interrupt, bytecount, data
247	F7	Define kerning points	17	Cap line, baseline, 6 x coordinate values
248	F8	Load three LUTs	770	Zero byte, three LUTs
249	F9	Virtual screen max dimensions	4	X & Ydimensions, max dimensions
250	FA	Output image dimensions	10	Zero byte, 4 coordinate values
251-4	FB-FE	Load LUTs individually	258	Zero byte, one LUT
255	FF	Null command	1	---

Table 4-6 List of SCODL Opcodes by Opcode Number

(0) and (255) Null Command

Using this command in your SCODL file will have no effect when the file is processed. It is used when data must be supplied only when it is expected.

For example, use this command if you need to define a macro character that is used in string commands such as a blank space. Opcode 227 (Define a macro) must be used in this case to provide the family/font and macro codes for the blank space character. Then, either opcode 0 or opcode 255 must be specified as the SCODL data making up the macro.

Byte #	Description
0	0 or 255 (x'00' or x'FF') - Null Command

Table 1 Null Command Format

(1) Single Edge

Table 1 below outlines how data is entered to describe a line called a *single edge*. A single edge requires a color code and two endpoints specified by the coordinate pairs (X1, Y1) and (X2, Y2). It is drawn one pixel wide.

Byte #	Description
0	1 (x'01') - Single edge
1	Color - Specifies the color of the edge
2 - 3	X1 coordinate
4 - 5	Y1 coordinate
6 - 7	X2 coordinate
8 - 9	Y2 coordinate

Table 1 Single Edge (Line) Command Format

For example, the following string of bytes shown in decimal and hexadecimal describes a line on the virtual screen similar to that shown in Figure 1 below:

(Dec.) 1 23 0 10500 12000 0 (Hex.) x'01 17 00 00 29 04 2E E0 00 00'

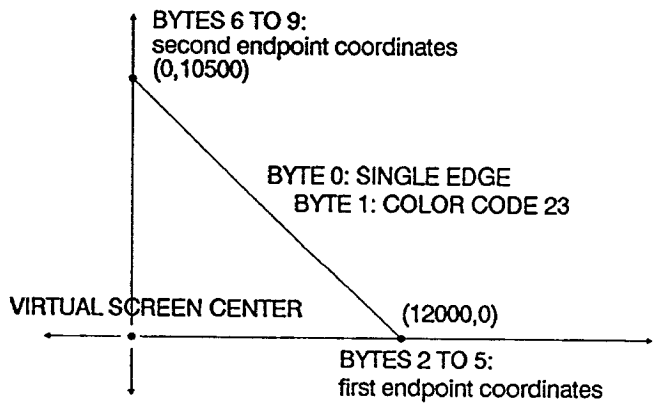


Figure 1 Example of a Single Edge

(2) Non-filled Arc

An arc is defined by two endpoints and an intermediate point and is taken from the three points that define the circle. This opcode can be used to produce a straight line by making the three points co-linear or to produce a circle by giving the initial and final points the same coordinates.

The intermediate point is placed approximately midway between the endpoints of the arc, to reduce arithmetic error in calculating the arc.

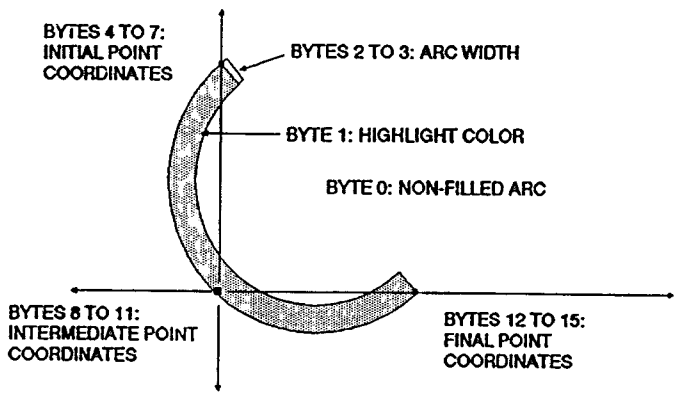


Figure 1 Example of a Non-filled Arc

An arc can be widened inward, toward its center. The color specified for the arc (byte #1) is applied to this widened boundary called the *highlight area.* Located at the ends of the arc is the arc's chord, which is a straight line between the endpoints.

The area inside the arc and the chord is called the *fill area.* A *non-filled* arc indicates that the fill area is transparent.

If the arc is very wide or shallow, the fill area may not appear as illustrated in Figure 2. Also, if the arc width specified is larger than the arc's radius it is treated as being the size of the radius.

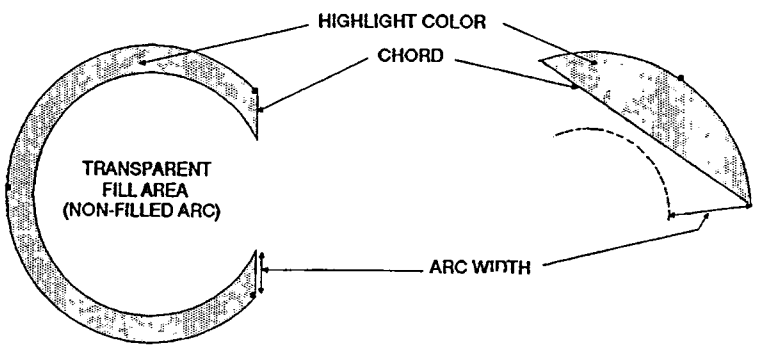


Figure 2 Definitions of Highlight and Fill Areas of Arcs

It is possible to describe arcs or circles that have sections extending past the virtual screen boundaries. Figure 3 shows such a case: all three arc defining points are within the virtual screen boundaries, but the actual arc defined has a section that is off the coordinate system. The arc is drawn, but the section off the screen does **not** appear.

> ***NOTE:*** *Off-screen arcs can not be included in opcode 227 (Define a macro).*

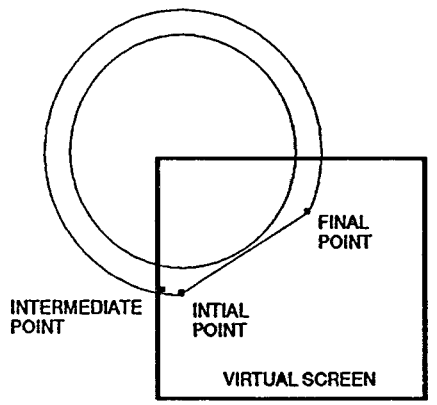


Figure 3 Defining an Arc Extending Off the Virtual Screen

Table 1 below illustrates the command format for the non-filled arc. Refer to opcode 3 (Filled arc) for a description of filled arcs. The arc is drawn from coordinate (X1, Y1) through (X2, Y2) to (X3, Y3).

Byte #	Description
0	2 (x'02') - Non-filled Arc
1	HC - Specifies the color of the arc
2 - 3	AW - arc Width
4 - 5	X1 coordinate
6 - 7	Y1 coordinate
8 - 9	X2 coordinate
10-11	Y2 coordinate
12-13	X3 coordinate
14-15	Y3 coordinate

Table 1 Non-filled Arc Command Format

Example: to draw a non-filled arc of color 23 and width 100 from (0, 10500) through (0, 0) to (12000, 0) the following string encoded in decimal and hexadecimal formats can be used:

(Dec.) 2 23 100 0 10500 0 0 12000 0
(Hex.) x'02 17 00 64 00 00 29 04 00 00 00 00 2E E0 00 00'

This creates the arc illustrated in Figure 1.

(3) Filled Arc

Table 1 below outlines the descriptor format for the filled arc. *Filled* indicates that the fill area between the inside of the arc boundary and the arc chord is filled in with the color specified by the third byte. Non-filled arcs have transparent fill areas through which any underlying objects are visible. As with non-filled arcs (opcode 2), the width of the arc boundary is determined by **AW** (arc width) and the color by **HC** (highlight color). The arc is drawn from the first to the third coordinate pair through the point defined by the second pair.

Byte #	Description
0	3 (x'03') - Filled arc
1	HC - Specifies the highlight color of the arc
2	0 - reserved
3	FC -Specifies the fill color of the arc
4 - 5	AW - arc width
6 - 7	X1 coordinate
8 - 9	Y1 coordinate
10-11	X2 coordinate
12-13	Y2 coordinate
14-15	X3 coordinate
16-17	Y3 coordinate

Table 1 Filled Arc Command Format

Example: To draw a filled arc of highlight color 23, fill color 56 and width 100 from (0, 10500) through (0, 0) to (12000, 0) the string expressed below in decimal and in hexadecimal is used:

(Dec.) 3 23 0 56 100 0 10500 0 0 12000 0
(Hex.) x'03 17 00 38 00 64 00 00 29 04 00 00 00 00 2E E0 00 00

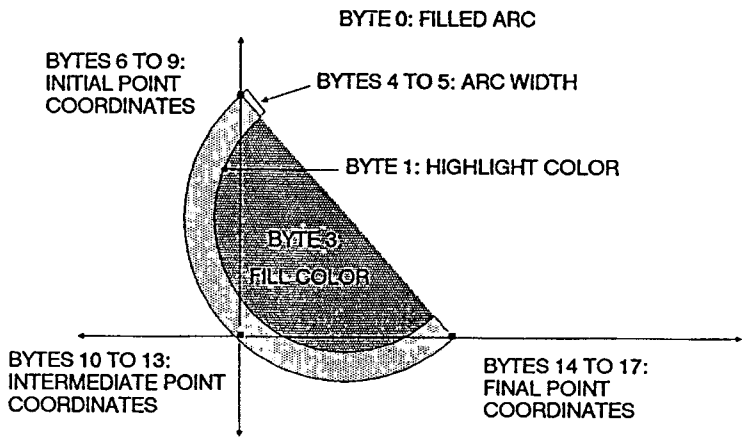


Figure 1 Example of a Filled Arc

(4) Polygon

A polygon is made up of a set of arcs and/or lines that describe its boundary. These arcs and/or lines are entered as a string of at least two coordinate points. The least significant bit (LSB) of coordinate data is not used for locating points on the virtual screen. The unused LSB of X coordinates is used in this opcode and in opcodes 5 and 6 to indicate whether the following points specify a line or an arc.

If the LSB of a coordinate is set to zero, it indicates that this coordinate and the following coordinate define a line. If the LSB is one, this coordinate and the next two coordinates define an arc. The final coordinates of any line or arc provide the initial point of the following line or arc and the polygon is closed by drawing a line or arc from the last point back to the first. Note that if the last edge in a polygon is an arc, only the initial and intermediate points need to be specified. The final point of the arc is assumed to be the first point in the list of vertices.

Assume we have the coordinate series:

X1, Y1, X2, Y2, X3, Y3, X4, Y4 ....

If the LSB of the first X coordinate is set to one, the first three points define an arc edge. Point three becomes the initial point of the next arc or line. If the LSB of the X coordinate of the third point is reset (0), the third and fourth points define a line.

A polygon definition requires a minimum of two points. A polygon description with only one coordinate pair produces undefined results and should not be attempted. The end of a polygon description is indicated by setting the last Y coordinate flag bit to a one. Intersection of edges of the same polygon is allowed only at the vertices. Polygon edges must not cross each other; the results are undefined if they do.

Table 1 below specifies the settings of a coordinate pair's flag bits.

Coordinate Pair LSB's			
X	**Y**	**Type**	**Description**
0	0	L	Initial coordinate pair of straight edge (line).
1	0	S	Initial coordinate pair of arc edge (spline).
0	1	E	Last coordinate pair (end point).
1	1	B	Reserved for future use (in this case, break).

Table 1 Coordinate Pair Flag Bit Assignments

Byte #	Description
0	4 (x'04') - Polygon
1	HC - Specifies the highlight color of the polygon
2	0 - reserved
3	FC - Specifies the fill color of the polygon
4 - 5	X1 coordinate
6 - 7	Y1 coordinate
8 - 9	X2 coordinate
10-11	Y2 coordinate
12-13	X3 coordinate
.	.
.	.
.	.
4n 4n+1	Xn coordinate (n = total number of coordinate points in polygon's boundary)
4n+2 4n+3	Yn - nth Y coordinate

Table 2 Polygon Command Format

The following string describes the polygon represented in Figure 1 with highlight color 23 and fill color 56:

(Dec.) 4 23 0 56 0 10500 1901 10500 4500 4500 10500 5000 12000 0 1 0 -3000 5501
(Hex.) x'04 17 00 38 00 00 29 04 07 6D 29 04 11 94 11 94 29 04 13 88 2E E0 00 00 00 01 00 00 F4 48 15 7D'

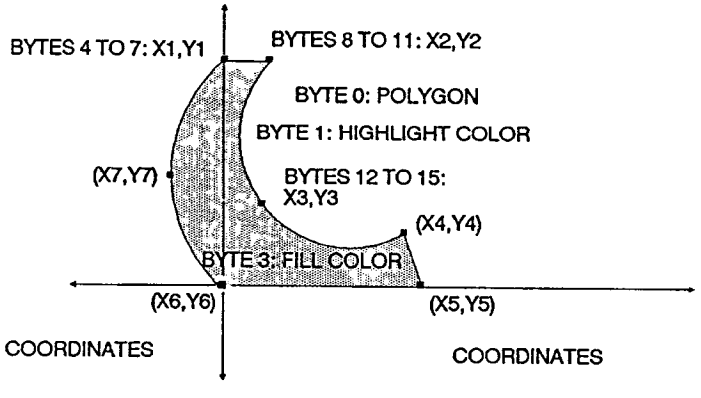


Figure 1 Example of a Polygon Comprised of Lines and Arcs

The oddness of the X coordinates of points 2 and 6 flags these points as the initial points of arcs. Because point 6 begins an arc and because point 7, the arc's intermediate point, is the last point given indicated by the oddness of the y coordinate. The arc is assumed to end back at point 1 and completes the polygon.

(5) Hole Polygon

This opcode creates a transparent area in previously drawn opcode 4, 5, or 8 polygons. It enables you to create holes required by macro characters such as **A, B,** and **R.** It must immediately follow an opcode 4 or 8 polygon or another opcode 5 polygon, with no other SCODL commands in between.

A polygon defined by this command appears as an outline in the same highlight color as the last preceding opcode 4 or 8 polygon. It also has the same fill color as this polygon, which shows wherever the opcode 5 polygon fails to overlap a polygon of the same priority written previously.

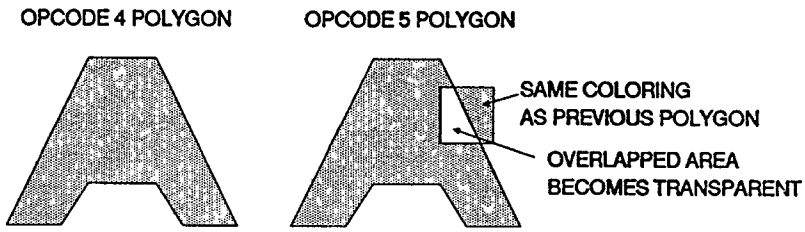


Figure 1 Behavior of a Hole Polygon

The hole polygon is an exception to the object priority rule outlined in Section 4, *Overlapping Priorities*. It is given the same priority level as an immediately preceding polygon, whether this is a regular polygon or another hole. It only acts as a hole on these objects. On lower priority areas it appears as a normal polygon and adopts the colors of the last normal polygon.

Where several polygons of the same priority exist, such as in the case where one normal polygon is followed by a set of hole polygons, holes are only created where the number of overlapping areas of the same priority is even. When this number is odd, the transparency effect is cancelled by the last layer and only the fill color set by the original normal polygon appears. This is demonstrated in Figure 2.

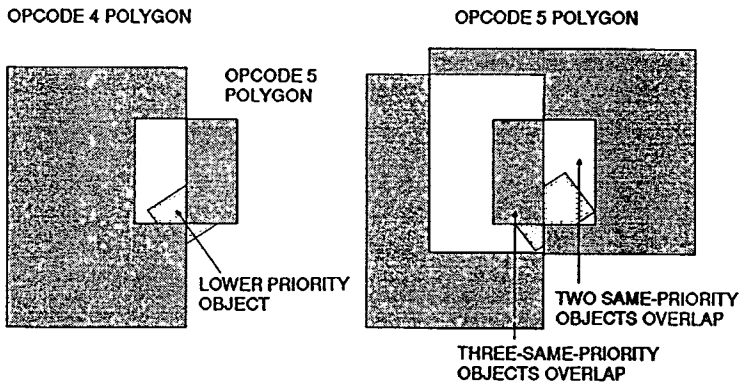


Figure 2 Behavior of Multiple Same-Priority Polygons

Notice that only the fill area of a hole polygon becomes the hole. The highlight area of the hole polygon is drawn in the color set by the original normal polygon.

Byte #	Description
0	5 (x'05') - Hole polygon
1 - 2	X1 coordinate
3 - 4	Y1 coordinate
5 - 6	X2 coordinate
7 - 8	Y2 coordinate
9 -10	X3 coordinate
.	.
.	.
.	.
4n-3 4n-2	Xn coordinate; n = number of coordinate points in the polygon
4n-1 4n	Yn - nth Y coordinate

Table 1 Hole Polygon Command Format

(6) Multiple Edge

A multiple edge description is made up of a set of connected arcs and line edges. The main difference between a multiple edge and a polygon description is that for a multiple edge the last coordinate pair specified is not automatically joined to the first to create an enclosed area as it is for a polygon description. Since there is not necessarily an enclosed area, there is no fill color. Also, a multiple edge may cross itself any number of times.

The coordinate data follows the same convention as that for polygons. The LSB of the X coordinate of the initial point of the arc or edge specifies whether the following points specify a line or an arc. If the LSB is zero, this coordinate and the following coordinate defines a line. If the LSB is one, this coordinate and the next two coordinates define an arc. The final coordinates of the previous arc or line define the initial point of the following arc or line.

The LSB of the initial point X coordinate has no significance as far as the coordinate location information is concerned; it is only a flag bit. The end of a multiple edge description is flagged by setting the last Y coordinate flag bit to a one. Table 1 listed in the description of Opcode 4, specifies the settings of a coordinate pair's flag bits.

A multiple edge definition requires a minimum of two points. A description with only one coordinate pair produces undefined results.

Byte #	Description
0	6 (x'06') - Multiple edge
1	HC - Specifies the color of the edges
2 - 3	X1 coordinate
4 - 5	Y1 coordinate
6 - 7	X2 coordinate
8 - 9	Y2 coordinate
10-11	X3 coordinate
4n-2, 4n-1	Xn coordinate; n = total number of coordinate points
4n 4n+1	Yn coordinate

Table 1 Multiple Edge Descriptor Format

For example, to describe the multiple edge represented in Figure 1, the following string is used:

(Dec.) 6 23 18000 0 1 10500 0 0 12000 1 0 10501
(Hex.) x'06 17 46 50 00 00 00 01 29 04 00 00 00 00 2E E0 00 01 00 00 29 05

In this string, the points described by (1, 10500) and (0, 10501) are at the same location. In the first instance, the 1 is added to the X coordinate to flag the start of an arc edge. In the second instance, the 1 is added to the Y coordinate to flag the last point of the multiple edge description.

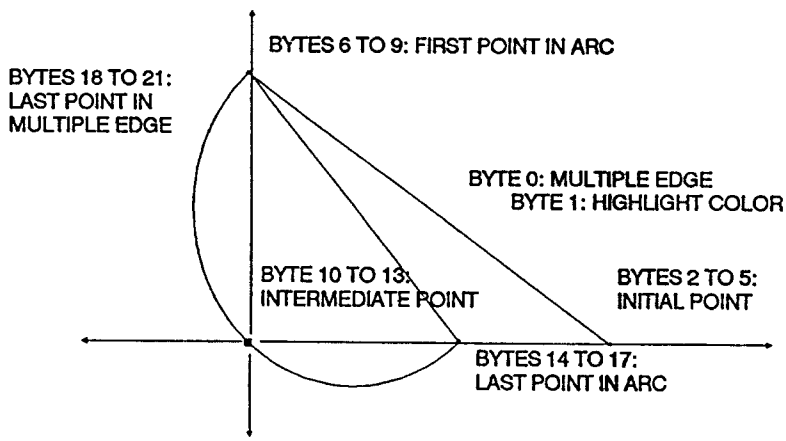


Figure 1 Example of a Multiple Edge

(7) Flood

This opcode allows an area of an image to be given a different color without respecifying the object or objects that define the area. The opcode specifies a single point and the color code of the new color.

The flood is performed on the image as it existed when the flood command was given. SCODL commands added after the flood command are treated normally. Therefore, flooding can be used anywhere in the file, including the beginning where it can change the background color.

Seed Points

The point, called the *seed point*, must be within the specified output image area. From this point, the new color floods outward until it encounters either a color boundary or the edge of the output image. The flood color spreads among all pixels that are the same color as the seed point as long as they are directly adjacent, and share one full side (regardless of whether they belong to highlight or fill areas of objects or the background area of the image). Pixels meeting at only one corner are not considered adjacent.

> ***CAUTION:*** *Flooding should be used sparingly, as it requires large computation and memory demands. If the MVP+ runs out of memory, flooding stops at whatever stage has been reached, and proceeds to the next command.*

If the RLC image to be flooded is in virtual memory, the flood command is ignored. In both cases, a **? 27** error is displayed. For recoloring a simple area, it is more efficient to use colored polygons than to perform flooding.

The location of the seed point is given as a pair of virtual screen coordinates. When more than one output resolution is used, the relation of this point to objects in the output image can change slightly between resolutions. Under these circumstances, placing the seed point within an accuracy of a few pixels can be difficult.

Therefore, if you vary your output resolutions, keep seed points away from boundaries unless you are required to do so. In the latter case, the selection of seed point coordinates should be based on virtual screen information rather than on output image coordinates.

Byte #	Description
0	7 (x'07') - Flood
1	Flood color
2 - 3	Seed point's x coordinate
4 - 5	Seed point's y coordinate

Table 1 Flood Command Format

In Figure 1(a), two filled arcs occupy the output image area of a SCODL image. The highlight area of one arc and the fill area of the other are the same color as the background of the image.

Therefore, a flood begun at the seed point indicated in part (b) of the figure fills all of these areas. As shown in (c), however, the output image boundary and the color boundary of one arc's fill area prevents the flood from reaching the area of background color at the upper right.

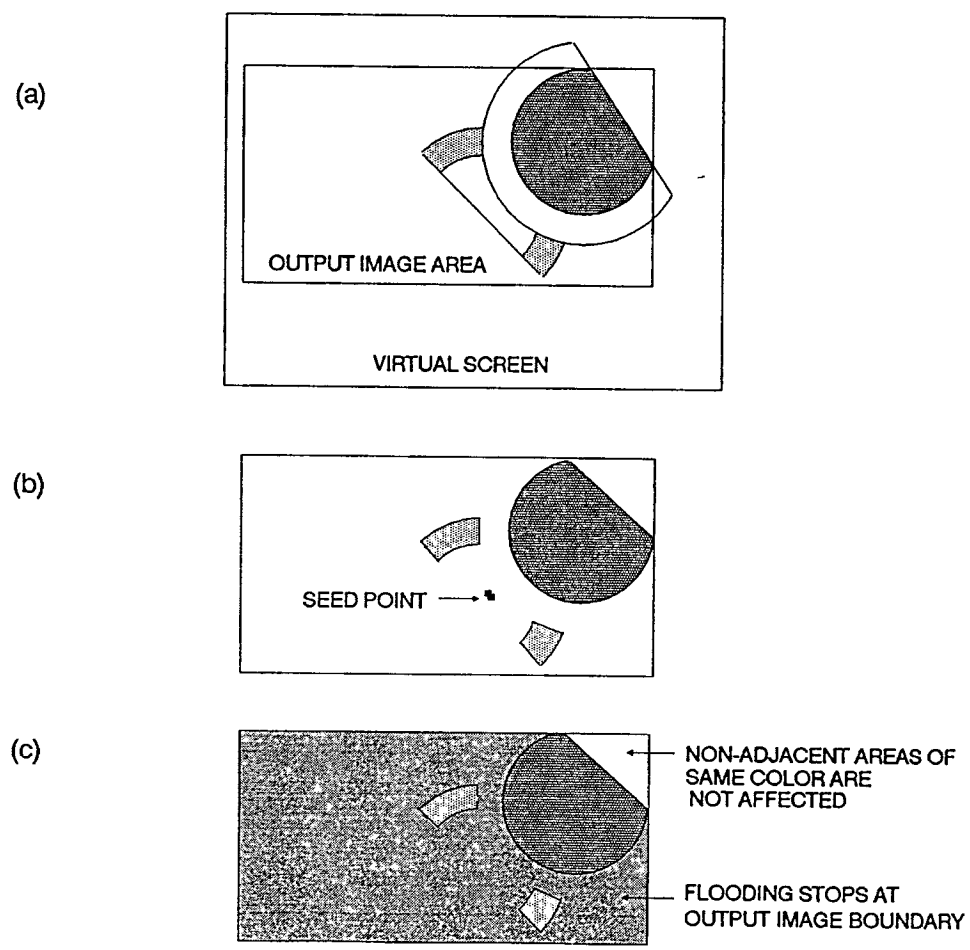


Figure 1 Example of Flood Color Behavior In an Image

(8) Compressed Mode Polygon

This command provides a way of defining a polygon in about half the number of bytes otherwise required, provided the polygon contains no arcs. It also allows you to produce a similar polygon elsewhere or in another size by using the same set of coordinate data.

Each coordinate in this object description is defined as a difference from the previous coordinate. This difference, or *differential*, is expressed with a single byte each for the vertical and horizontal differentials. Hence, each coordinate point is located with two bytes instead of four. Negative numbers may, if desired, be encoded in 8-bit 2's complement form.

The first pair of these differential coordinates must have a point of reference from which the point they describe can be located. This is provided by Opcode 243 (Specify Macro Base Address), which must be used with this command. Changing the base address changes the location of the compressed mode polygon on the virtual screen.

> ***NOTE:*** *If the compressed polygon is being included in a macro definition, the base address does not apply to the virtual screen on which the macro is defined.*

Within the macro, the base address for the polygon is considered to be the screen center (0, 0). The macro base address specified by opcode 243 can be used to position the macro box within the larger image.

The Weighting Function

To add this polygon to the image, SCODL must translate the differential values into absolute virtual screen locations by adding the first pair of differential values to the base address and continuing to add each subsequent differential pair to the preceding location.

This process includes a weighting function that multiplies the differential distances by a factor determined by the value *w*. Changing *w* changes the size of the polygon when it is added to the image. You can adjust *w* using Opcode 235 (Specify Compressed Polygon Weighting) where *w* is a value between 0 and 8 inclusive. A value of 0 does not expand the polygon at all. A value of 8 expands the distances between coordinates by a factor of 2 to the 8 or 256 times.

SCODL expands the differential coordinates into coordinate addresses within the virtual screen by applying the following equations:

$$xc = xp + 2^{w} * xd$$
$$yc = yp + 2^{w} * yd$$

Where the characters represent:

xc & yc computed coordinate pair of the current polygon vertex
xp & yp coordinate pair of the previous polygon vertex
xd & yd differential coordinate pair for the current polygon vertex
w weighting function applied to each differential coordinate pair defined by Opcode 235

The polygon colors are determined using Opcode 242 (Specify Macro Colors). The end of the polygon is marked by a last pair of coordinates set to zero. The polygon automatically closes by a line drawn from the last non-zero coordinate pair to the first pair.

Byte #	Description
0	8 (x'08') - Compressed mode polygon
1	First differential X coordinate
2	First differential Y coordinate
3	Second differential X coordinate
4	Second differential Y coordinate
.	. . .
.	. . .
2n-1	nth differential X coordinate (n = total no. of coordinate points in polygon)
2n	nth differential Y coordinate
2n+1,2n+2	end-of-polygon flag (x'0000')

Table 1 Compressed Mode Polygon Command Format

In Figure 1 the same compressed mode coordinates have been used with different base addresses and weighting functions (opcodes 235 and 243) to create two different polygons. These polygons are described by the following strings:

(a) 243 2000 1000 242 26 235 0 0 8 (differential coordinates) 0 0
(b) 243 0 0 242 26 235 0 1 8 (differential coordinates) 0 0

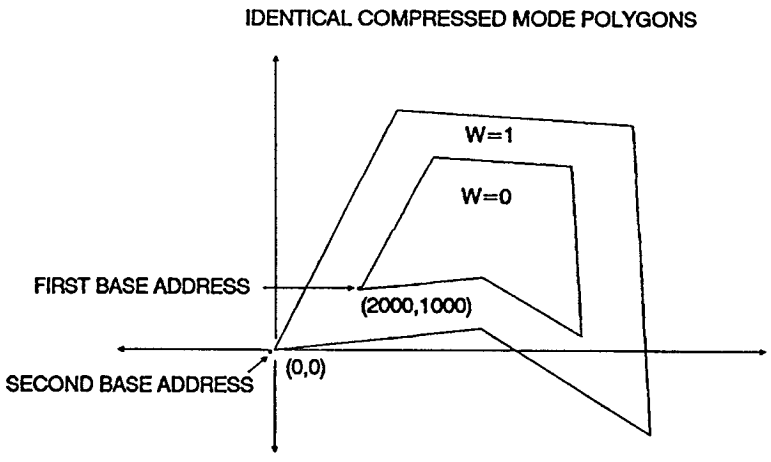


Figure 1 Example of the Use of a Compressed Mode Polygon

If the same base address is used, the starting points in the lower left corners of the two polygons coincide. If the same weighting function is used, both polygons will be the same size.

Note that since the weighting function only expands and does not shrink a polygon, the polygon coordinates must have been defined at the smallest size at which you plan to draw the polygon.

(9) Polygon With Spline Curves

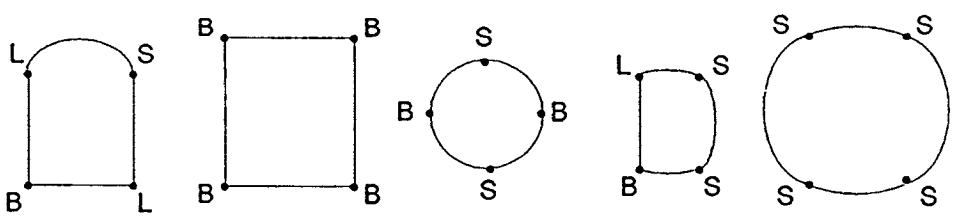
This opcode is similar to opcode 4 (Polygon) except that it incorporates a function that draws a smooth, continuous curve called a *spline* through any series of more than three points.

Recall that in an opcode 4 polygon, the LSBs of the coordinate values are used to indicate whether a point begins a straight edge or begins an arc. In opcodes 9, 10, and 11, these bits indicate one of the following: a point that continues a straight line or continues a spline curve from the previous point; a *break point* from which either a straight line or spline curve may begin regardless of what function preceded the break point; and, as in an opcode 4 polygon, the last point in the object is indicated if its Y coordinate flag bit is set to a one. Refer to Table 1.

Coordinate pair LSB's			
X	**Y**	**Type**	**Description**
0	0	0	Continuation of straight line from previous point
0	1	1	End point, or last coordinate pair
1	0	2	Continuation of spline curve from previous point
1	1	3	Break point: the preceding function ends. This point may be followed by either a straight edge or a spline curve.

Table 1 Coordinate Pair Flag Bit Assignments

Examples (these are the same for opcodes 10 and 11 also):



Notes:

1. For closed shapes, as in all cases of opcodes 9 and 10 and some cases of opcode 11, the choice of a starting point does not affect the resulting shape.
2. Splines for which only three points are given are treated as regular arcs.

Byte #	Description
0	9(x'09') - Polygon with Spline Curves
1	HC - Specifies the highlight color of the polygon
2	0 - reserved
3	FC - Specifies the fill color of the polygon
4-5	X1 coordinate
6-7	Y1 coordinate
8-9	X2 coordinate
10-11	Y2 coordinate
12-13	X3 coordinate
.	.
.	.
.	.
4n 4n+1	xn coordinate; n = total number of coordinate points in polygon's boundary
4n+2 4n+3	Yn - nth Y coordinate
4n+4	Specifies type of nth coordinate, either L, S, or B (see Table 1). The last coordinate point will be treated differently depending on the value of this byte.

Table 2 Polygon with Spline Curves Command Format

(10) Hole Polygon With Spline Curves

This opcode is similar to opcode 5 (Hole Polygon) except that it incorporates a function that draws a smooth, continuous curve called a *spline* through any series of more than three points. Splines that are given only three points are treated as regular arcs.

This opcode must immediately follow an opcode 4, 5, 8, 9, or 10 polygon with no other SCODL commands in between. A polygon defined by this command appears as an outline in the same highlight color as the last preceding opcode 4, 8, or 9 polygon. It is subject to the same overlapping and fill color behaviour described for opcode 5.

Byte #	Description
0	10 (x'0A') - Hole Polygon with Spline Curves
1-2	X1 coordinate
3-4	Y1 coordinate
5-6	X2 coordinate
7-8	Y2 coordinate
9-10	X3 coordinate
.	.
.	.
.	.
4n-3 to 4n-2	Xn coordinate; n = total number of coordinate points in polygon's boundary
4n-1 to 4n	Yn - nth Y coordinate
4n+1	Specifies type of nth coordinate, either 0, 2, or 3. Refer to Table 1 in the description of opcode 9. The last coordinate point is treated differently depending on the value of this byte.

Table 1 Hole Polygon with Spline Curves Command Format

(11) Multiple Spline Edge

A multiple spline edge description is made up of a set of connected line edges and/or splines. As for a multiple edge, the last coordinate pair specified is not automatically joined to the first to create an enclosed area. Since there is not necessarily an enclosed area, there is no fill color. Also, a multiple spline edge may cross itself any number of times.

Byte #	Description
0	11 (x'OB') - Multiple spline edge
1	0 - open
	1 - closed
2	HC - Specifies the color of the edge
3-4	X1 coordinate
5-6	Y1 coordinate Note: The first point is treated as a break point regardless of its flag bits.
7-8	X2 coordinate
9-10	Y2 coordinate
11-12	X3 coordinate
4n-1, 4n	Xn coordinate; n = total number of coordinate points
4n+1, 4n+2	Yn coordinate
4n+3	Specifies type of nth coordinate, either L, S, or B. Refer to Table 1 in the description of opcode 9. The last coordinate point is treated differently depending on the value of this byte. If this byte identifies it as a spline point (2), it is treated as a break point (3).

Table 1 Multiple Edge Descriptor Format

The coordinate data follows the same convention as that for opcodes 9 and 10. The LSBs indicate whether the point continues a straight line or continues a spline curve from the previous point or they may indicate a *break point* from which either a straight line or spline curve may begin, regardless of what function preceded the break point. Refer to Table 2.

Coordinate pair LSB's			
X	Y	Type	Description
0	0	L	Continuation of straight edge (line) from previous point
0	1	E	End point, or last coordinate pair in multiple spline edge
1	0	S	Continuation of spline curve from previous point
1	1	B	Break point; preceding function ends. May be followed by a straight edge or a spline curve.

Table 2 Coordinate Pair Flag Bit Assignments for Multiple Spline Edge

(128) End Of Image Description Marker

This opcode signifies that all object descriptions describing an image have been read in and is followed by a zero byte. When this command is received, MVP+ processing of the image begins. When you intend to perform RLC merging, this command should not be sent until the RLC data (opcode 219) and its dimensions and transparency color (opcodes 220 and 221) have been given. This command can be used alone to generate a blank image.

When a QCR is configured as the output device, this command causes the QCR to advance its film and perform an automatic intensity calibration unless automatic performance of either or both of these operations has been disabled.

Byte #	Description
0	128 (x'80') - End of image marker
1	0 - zero byte

Table 1 End of Description Marker Command Format

(129) SCODL Status Report Request

During configuration of the MVP+, you have the option of enabling or disabling a mode in which the MVP+ sends a SCODL status report after each image. If no option is specified, the MVP+ sends the report after each image that is successfully completed. This opcode enables you to turn this mode off or on without reconfiguring the MVP+. It also allows you to request an immediate SCODL status report.

The SCODL status report is a string of bytes containing encoded information about SCODL. Some of this is information about the modes in which SCODL has been placed by your use of configuration options and opcodes such as the output image locators and the number of user-defined macros currently stored. Other information describes the SCODL processing status, such as the amount of the current image processed so far and the frame number of the last processed image.

Table C-1 in Appendix C, *The SCODL Status Field*, outlines how the bytes in the report are divided into fields and describes each of these fields.

Byte #	Description
0	129 (x'81') - SCODL Status Report request
1	1 - Return SCODL status immediately.
	2 - Turn on the after-image return of SCODL status.
	3 - Turn off the after-image return of SCODL status.

Table 1 Status Report Request Command Format

(130) Return Macro String Length

This command returns dimensions of the supplied string. The dimensions are returned in a similar fashion to the status returned by opcode 129 (SCODL Status Report Request). The command is of the form:

Byte#	Description
0	130 (x'82') - return string dimensions
1	MODE BITS 7 = 0 return string width & height = 1 return enclosing rectangle coordinates 6 = 0 single-byte character string supplied = 1 double-byte character string supplied
2-n	n-1 bytes of character data in the supplied string
n+1	x'FF' - end of string delimiter

Table 1 Return Macro String Length Command Format.

Currently only Mode 0 is supported, returning the width and height of a single-byte character string. In this mode, the MVP+ returns a message string in an ASCII string of hexadecimal codes. The format of the returned status is:

#.00..wwwwhhhhbbbbcccc

Where the following characters represent:

#	Delimiter indicating non-status data being returned space character
wwww	Hexadecimal width of the scaled string in SCODL coordinate format. Binary point between bit 13 & 12.
hhhh	Height of the string
bbbb	Distance from the macro center to the base line
cccc	Distance from the macro center to the cap line

(206) Macro Access Mode

This command specifies whether the SCODL processor is to get its macro descriptions from its own internal memory or from virtual memory. Before virtual memory-based macros can be used, they must be installed using the program VMACNSTL.EXE.

All macros stored in virtual memory use a JIS code for identification. Depending on the mode you select, virtual memory-based macros may be accessed by either their JIS code or an equivalent Word Processing code. The relationship between Word Processing codes and their JIS equivalents is found in the NIS publications *First Level Kanji Code List* and *Second Level Kanji Code List*. When the Word Processing mode is selected, the MVP+ translates all macro requests into JIS codes before going to virtual memory.

Byte #	Description
0	206 (x'CE') - Select macro access mode
1	Access mode 0 - Access macro descriptions found in MVP+ memory. This is the default mode. 1 - Macro descriptions are to be found in virtual memory, you must supply JIS codes. 2 - Macro descriptions are to be found in virtual memory, you must supply Word Processing codes. 3 - Codes stored as normal macro codes.
2-3	Font Pointer Table font selection code. When a set of macros is installed in virtual memory, it is assigned a font selection code (0-255).

Table 1 Macro Access Mode Command Format

The Font Pointer Table allows up to 256 entries. The first 3 entries are used as follows:

FPT 0 - KANJI font in JIS selectable format
1 - Western fonts
2 - KANJI rotated characters for Kanji font 0
3 - Mincho
4 - Gothic

Western fonts (macro access mode 3) are stored in groups of 256 characters/font. The supported Western fonts are as follows:

0 - MVP internal, sans-serif
1 - Times
2 - Times Bold
3 - Triumvirate
4 - Triumvirate Bold
5 - Swiss - sans serif
7 - Swiss Bold - sans serif
17 - Dutch Roman - serif
19 - Dutch Bold - serif

(212) RLC Mapping Data

This command supplies RLC data to be mapped into a SCODL virtual screen image. This data is located according to the parameters specified by opcode 213. It's strings of pixels are translated into SCODL objects when they are added to the image.

Currently, only RLC data that is in the IBM PGA format for RLC data can be used in this command. This data format is described in Table 1 below.

> ***NOTE:*** *The use of this command inside a macro definition is not currently supported.*

Byte #	Description
0	212 (x'D4') - RLC mapping data
1	1 - indicates IBM PGA data format. No other formats are supported yet.
2 to N+1	N bytes of RLC image data. The RLC format is indicated in Table 2 below.
N+2, N+3	x'00FF' - end-of-RLC data marker

Table 1 RLC Mapping data Command Format

IBM PGA data format:

Byte #	Description
0	For SCODL purposes, this can be any non-zero value For IBM PGA purposes, this must be one of two commands: IMAGER (Image Read - hex value x'D8') or IMAGEW (Image Write - hex value x'D9')
1-2	Line number (must be one "flipped" word - low byte first)
3-4	X coordinate (from left edge) of starting pixel (must be one "flipped" word - low byte first)
5-6	X coordinate (from left edge) of ending pixel (must be one "flipped" word - low byte first)

> ***NOTE:*** *For RLC image data, the origin (0, 0) is the lower left corner of the image*

Table 2 IBM PGA data format for RLC data (continued...)

This data is followed by any number of "PEL packets". These are basic units of RLC data in one of two forms:

Run Length Form,
for areas of solid color: 1 byte "N" for pixel count of N+1 (N=0 to 127, count = 1 to 128)
1 byte color code

Pixel Form,
for areas of mixed color: 1 byte "N" for pixel count N+1 -128 (N = 128 to 255, count = 1 to 128).
Note that count must be 128 to 255)
N+1 - 128 color code bytes, each describing one pixel

Table 2 IBM PGA Format for RLC Data

(213) RLC Mapping Parameters

This command specifies the virtual screen area the RLC data, provided by opcode 212, is to be mapped to. It also provides the parameters of the data, including the number of pixels in each scan line and the number of scan lines.

RLC mapping does not require a particular relationship between the parameters of the RLC data and the virtual screen area it is to fit into. Because the RLC data is translated into SCODL objects, it can be enlarged or compressed in either or both dimensions to fit the target area on the virtual screen.

Byte #	Description
0	213 (x'D5) - RLC mapping parameters
1	0 - unused
2-3	VX1 - leftmost virtual screen X coordinate
4-5	VX2 - rightmost virtual screen X coordinate
6-7	VY1 - topmost virtual screen Y coordinate
8-9	VY2 - lowermost virtual screen Y coordinate
10-11	XPIX - maximum number of pixels per scan in RLC data
12-13	YPIX - maximum number of scan lines in RLC data

Table 1 RLC Mapping Parameters Command Format

RLC merging and RLC mapping are two ways of placing previously stored RLC data within a SCODL image. Merging must be performed in RLC storage mode. The RLC's are actually merged into the output image, rather than onto the virtual screen.

In RLC mapping, no transparent areas are created. RLC's are mapped onto the virtual screen rather than onto the output image. In the process of mapping, each RLC becomes a SCODL polygon. This means that regardless of the size of the target area given by the parameters above, the overlaid data shrinks or stretches in either or both dimensions in order to fit.

Because of this feature, RLC mapping is most likely to be used simply as a means of processing a relatively small RLC image into a full-sized hard copy. This can be done by setting the parameters VX and VY, as well as the output image dimensions, to the dimensions of the output image area (see opcode 250) of the virtual screen.

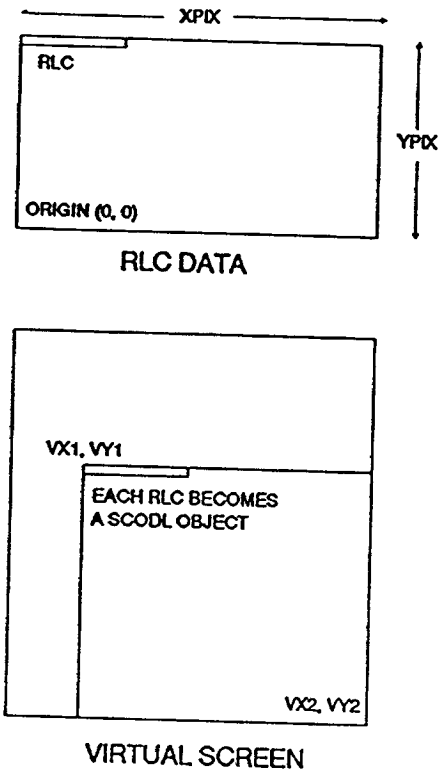


Figure 1 RLC Mapping Parameters

(214) Specify a Kerning Table

The description of opcode 247 (Set Kerning Points) explains how each macro character can be assigned a set of kerning point values. Kerning points then determine the Inter Character Spacing (ICS), measured when two characters are placed side by side in a string. See opcodes 233 (String Spacing) and 247 for more information.

When outputting, certain specific pairs of characters may not end up being spaced satisfactorily, even when care has been taken to establish appropriate kerning point values. This command lets you specify a combined width value for specific character pairs, replacing the combined width determined by the regular kerning process when those two characters appear together in that order.

In the regular kerning process, SCODL compares the three pairs of kerning values between the characters and finds the pair - top, middle, or bottom - adding up to the largest value and the ICS specified by opcode 233. This gives a total distance measured as a standard SCODL fraction between the base addresses of the two characters.

Before proceeding, SCODL checks the kerning table loaded by this command for a listing of the character pair. If there a listing, the combined width value is taken from the table and the ICS specified by opcode 233 is added, not the largest pair of kerning values.

Byte #	Description
0	214 (x'D6') - Specify a Kerning Table
1	Family/font code
2	0 - reserved
3-4	N - Number of entries in the table (number of character pairs with supplied combined widths). $0 < N < 16380$
5	Macro code of left character of first pair
6	Macro code of right character of first pair
7-8	Combined width value to be used for first pair
Nx4+1	Left character of Nth (last) pair
Nx4+2	Right character of Nth (last) pair
Nx4+3, Nx4+4	Combined width value to be used for Nth (last) pair

Table 1 Specify a Kerning Table Command Format

Example: Suppose the two characters RV shown in Figure 1 are requested together in a kerned string. Clearly, the topmost pair of kerning values represented in Figure 1 as RHS 1 of the first character and LHS 1 of the second character, add up to the largest combined width since they approach each other more closely than either the middle or bottom pairs of values.

If no kerning table exists for this font, or if such a table contains no listing for this particular character pair, RHS 1 and LHS 1 (say 0.3 and 0.5*) are added together with the ICS value given in opcode 233 (say 0.1). The resulting value of 0.9 in the standard SCODL data format is the distance between the characters' base addresses.

If you are dissatisfied with the results and want the characters farther apart, use a kerning table that specifies a combined width value for the character pair **RV**. To widen the pair, the combined width value should be larger than the sum of RHS 1 and LHS 1 (say 0.85).

Now, when **RV** is requested in a string, the combined width value in the kerning table overrides the regular kerning values for those characters. The ICS (0.1) is added to 0.85, instead of to 0.8, and the characters' base addresses is separated by a total distance of 0.95.

> ***NOTE:*** *These values represent character widths as fractions of the virtual screen (total width = 2), before any macro scaling is applied. The macro scale is a number of 1 or less that is multiplied by these values when the macro is added to the image.*

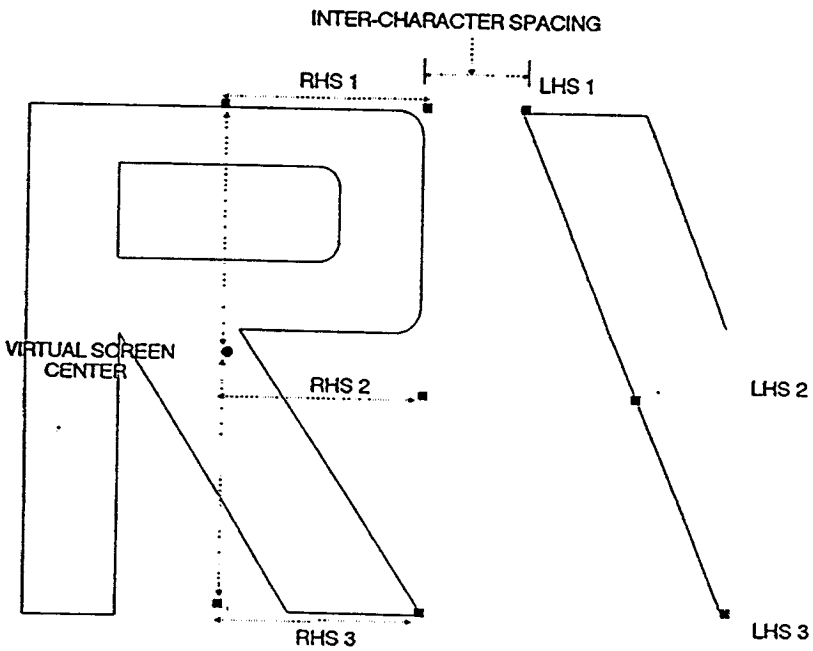


Figure 1 Regular Kerning: Kerning Point Values and ICS

(215) Select Type of LUT Compensation

This command prepares a LUT compensation that is performed if subcommand 1 of opcode 211 is received. It combines any currently loaded look-up table file with any other set of LUTs. The purpose is to adapt the current LUTs for irregularities in the color response of the output device. Therefore, this command should specify a second set of LUTs that is already distorted to compensate for these irregularities. When the LUTs are combined, the compensation function of the second set of LUTs is incorporated into the currently loaded set.

Because compensated LUTs are stored internally in output devices such as the QCR and PCR, subcommand 1 lets you specify one of these by its one-byte code. For example, the QCR contains LUTs that are compensated for 35mm Ektachrome 100 ASA color slide film as well as two kinds of Polaroid sheet film. Any currently loaded set of LUTs can be adjusted for the film response of one of these film types by using subcommand 1 and specifying the code given in the *User's Guide to the QCR* for the pre-adjusted compensation LUT.

Duration

The compensation specified by this command can not actually be performed until opcode 211 (LUT Compensation Control) turns it on. The resulting compensated LUT remains in effect for subsequent images unless: opcode 211 turns it off; new LUTs are loaded via SCODL opcodes 225, 248 or 251-54; or the MVP+ is restarted. The on/off selection of this command by opcode 211 stays in effect until another opcode 211 or until the MVP+ is restarted.

Background on LUTs

A set of look-up tables consists of three tables, each of which designates 256 color codes to 256 different levels of light intensity in each of red, green, and blue. A color is produced when the three light intensities for one of the codes are combined. The simplest use of LUTs is by using a *linear ramp* through the range of colors.

The compensated LUTs stored in the QCR simply compensate this ramp for a certain film type so that output colors do not adopt the greenish or bluish bias characteristic of some types of film.

However, a simple ramp through the range of possible colors is not appropriate for some of your drawing needs. For example, if you are drawing facial portraits you would want a palette that offers closely differentiated shades of brown and flesh tones and could sacrifice selection in other hues. If you have prepared such a palette, you would want it compensated against the color biases of the output system. This command provides such compensation.

Prior to the development of this command, if you had LUTs that represented a customized palette, you had to perform a linearization process as described in the *User's Guide to the QCR*. This requires comparing each of the LUT values with the output result using a densitometer and then adjusting each one to

compensate for the distortion caused by the output system. This command takes a LUT that is already linearized and combines it with the one that contains your own palette of colors.

Subcommand 1

This subcommand compensates using a LUT stored in an output device

Byte #	Description
0	215 (x'D7') - Select LUT Compensation
1	1 - Subcommand byte
2	Output device LUT number; a code number for a stored LUT which compensates for a certain output medium. For film recorders, use codes given in the description of command #35. Code x'FF' compensates using the LUTs currently loaded in the film recorder.

Table 1 Subcommand 1 Command Format

Example: If you use a QCR with a 35mm film module, you might initially let the QCR load the default LUTs for that module. The QCR identifies the module electronically, assumes that Ektachrome 100 is being used and loads LUTs for the specified resolution.

Suppose you are using Ektachrome 100 film and you want.to develop LUTs. You need a color palette that allows red, green, and blue combinations for each color code, but are not compensated for film response. You can combine your palette with the QCR's Ektachrome compensation LUT, assuming a 2K resolution, as follows:

1. Load the palette LUT using either opcode 248 or 251-4.
2. Send the command string 215 1 3 (Hex.: D7 01 03) to specify the Ektachrome LUT's compensation function.
3. Use opcode 211, subcommand 1 to have the compensation performed.

You may switch to Polaroid 559 film by using a 2 or a 5 in the second byte in Step 2 and applying it to the original LUT.

Figure 1 illustrates how this compensation works on the Blue third of the palette. The film's responses to intensities of Green and Red also deviate from the ideal. The Green and Red listings in the already-compensated QCR LUT would be used to adjust the Green and Red components of your palette in the same way.

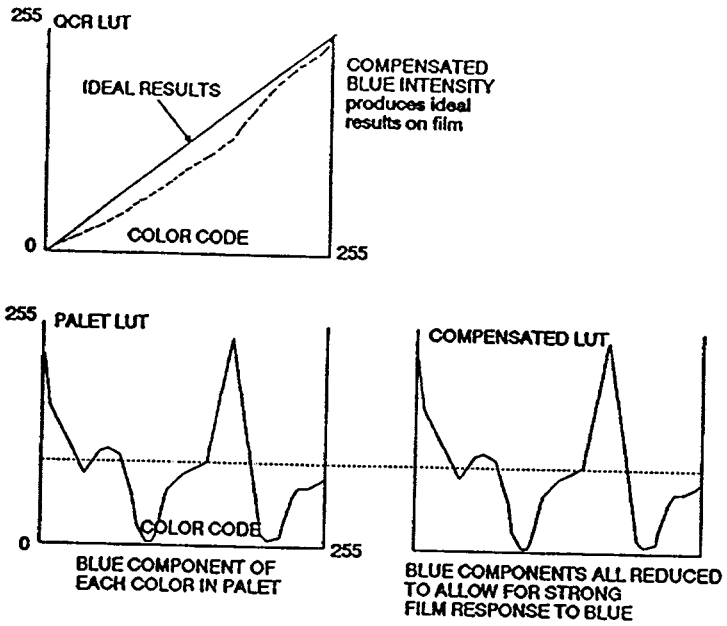


Figure 1 Example of the Blue Third Palette Compensation

Subcommand 2

This subcommand compensates using your own LUT.

As an alternative, subcommand 2 lets you load a compensation LUT of your own from outside the system (option 1) or reuse a compensation LUT previously loaded (option 0).

Byte #	Description
0	215 (x'D7') Select LUT Compensation
1	2 - Subcommand byte
2	0 - Use current compensation LUT - Use this option if you previously loaded a compensation LUT using the following option and have switched to subcommand 1, but wish to use the same compensation LUT again. If no LUT is in memory such as after startup, a linear LUT is used for compensation; no real compensation occurs.
	1 - Use the following compensation LUT
3-770	768-byte LUT to be used for compensation - sent if previous byte is option 1.

Table 2 Subcommand 2 Command Format

(216) Thicken Edge

This opcode causes any lines, arcs, multiple edges, or polygon boundaries that follow it, including those in macros, to be expanded to a thickness specified in the command. Thickening is not performed on edges produced from opcodes 2 or 3. Thickness stays in effect until changed by a subsequent opcode 216 or until a system reset is performed.

The desired thickness is stated as a standard two-byte SCODL coordinate value. Thickening is performed on the virtual screen. A thickness of 0 indicates no thickening. The thickness value *d* applies to the width of a polygon drawn around the line or arc as shown in Figure 1.

The second byte in the command specifies one of four ways to treat the end of the thickened area. Figure 1 shows thickening with option 0 (plain endpoints). Extended endpoints are shown in Figure 2.

Byte #	Description
0	216 (x'D8') - Thicken edge
1	Subcommand byte: 0 = plain endpoints
2 - 3	Thickness value; default = 0, indicating no thickening

Table 1 Thicken Edge Command Format

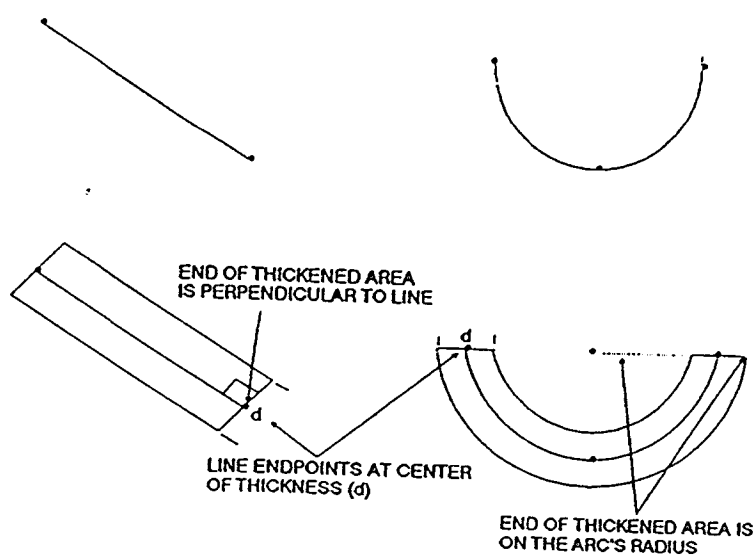


Figure 1 Examples of Line and Arc Thickening without Extended Endpoints

Any given value *d* produces a constant output thickness as long as the size and resolution of the output image are not changed. This relationship is governed by the formula:

$$d = N \times 65536/MAXDIM$$

where:

N is the number of output pixels thickness; MAXDIM is the larger of the numbers of pixels and lines in the output image called XDIM and YDIM. Refer to Section 5, *Output Image Selection* for more detailed information.

For example, suppose you want thickened lines and arcs to be eight output pixels wide in an output image with maximum dimension of 2048 output pixels. The value for *d* that results in at least this much thickening is:

$$8 \times 65536/2048 = 8.0 \times 32 = 256$$

This is the decimal mode SCODL coordinate.

Precautions

Because thickening a single edge causes a polygon with four vertices to be drawn, image complexity increases rapidly with the use of thickening. A likely consequence of the extensive use of thickening is that the MVP+ returns a **? 04** error message, indicating that the SCODL table called NCHKSLT has to be enlarged (opcode 239).

The most complex polygon that can be thickened is one of 65,535 sides, including the sides of all hole polygons (opcode 5). Larger polygons are left with some sides unthickened. If the requested thickening of an object results in part of the thickened area being off the virtual screen, thickening will not be performed.

Lines or arcs in macros may be thickened by adding this opcode before the macro request. They should not be thickened to the extent that they reach off the virtual screen they were defined on; the results will be undefined if they do. Note that this is especially likely to happen when thickening small, heavily scaled macros at low output resolutions.

If thickening is already in effect from a prior image at the start of an image file, and a new thickening value is going to be specified during that file, the value already in effect must be specified at the start. This is because in certain modes, the meta-file reads more than once. If the file does not begin with opcode 216, the thickening value in effect at the end of the file may be applied to the start of the file on the second reading.

Extended Endpoints

Subcommands 1, 2, and 3 specified by byte #1 in the command format extend the thickened area beyond the line or arc endpoints. This can be used to improve the appearance of vertices in a thickened polygon or multiple edge.

Each subcommand supports a different kind of end extension as illustrated in Figure 2. In all cases, thickening is extended beyond the endpoint to a distance equal to *d/2*, or *r*.

> ***NOTE:*** *When using subcommand 2 (angle endpoints), best results are achieved if the thickness in output pixels (***N** *in the equations above) is an odd number.*

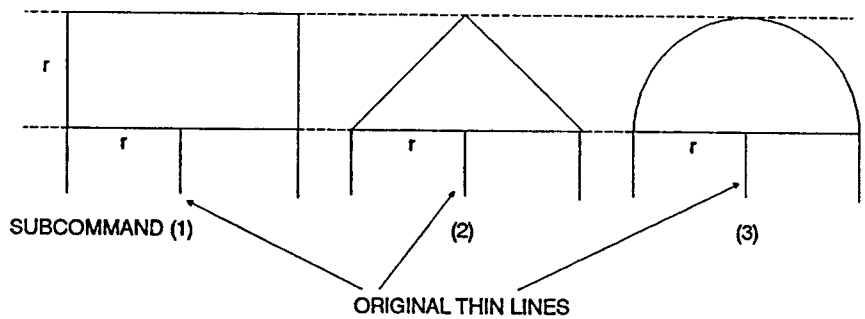


Figure 2 Three Different Extended Endpoint Styles

Thickening Arcs

In all cases, the extended endpoint begins at the radius of the arc. For rectangular endpoints (subcommand 1), the extended endpoint is a rectangle with one side on the radius of the arc.

When using one of these subcommands on an arc, which is nearly a complete circle, the extended endpoints of the same arc may overlap each other. If they do, the overlapped area becomes transparent.

(217) Specify Font Attributes

This opcode lets you define a consistent typographic *base line* and *cap line* for a macro family or font. These are positive distances above and below the center of the virtual screen on which each macro in the font is defined. They are labelled X-base and X-cap in the description of opcode 233. This opcode must appear before any object descriptors in the file.

Rather than defining the actual top and bottom of the individual character, these two horizontal lines provide a set of text alignment points that, when assigned to a string of characters via opcode 233, places the characters in correct typographic alignment.

To achieve this effect, use this command to specify cap and base lines for the whole macro family or font. Then, when defining individual characters, place *descenders* - the tails found on lower case g, y, q, p, j - below the base line and let long *ascenders* - the tops of upper case letters and of lower case b, d, k, l, h - rise to the cap line or to a consistent distance above or below it. Shorter ascenders - the dots of lower case j and i and the top of lower case t - rise an appropriate portion of this distance.

The SCODL internal character set bases X-cap and X-base on the top and bottom of the upper case X. It allows ascenders and descenders that are curves, such as the top of the upper case S and the bottom of the lower case g, which are its highest and lowest-reaching characters, to extend slightly further from screen center than those that are the ends of vertical lines.

> ***NOTE:*** *Since this font information is stored as a macro, the count of current macros given in the SCODL status report is incremented by one when this command is used.*

Byte #	Description
0	217 (x'D9') - Font attributes
1	Family/font code (1 - 127)
2	0 - reserved
3- 4	X-cap - font cap line; default is 0 indicating that the macro is defined on the top of the virtual screen.
5 - 6	X-base - font base line; default is 0 indicating that the macro is defined on the bottom of the virtual screen.

Table 1 Specify Font Attributes Command Format

(218) Specify Background Color

The background color in a SCODL image is the fill color of a rectangle the size of the virtual screen, which is drawn at the start of an image. The default color for this rectangle corresponds to color code 0. This command allows any color code from 0 to 255 to be assigned to this rectangle.

This command remains in effect until the MVP+ is restarted or another opcode 218 is received. An alternative way to change the background color for multiple images is to change the LUT values associated with color code zero. You can change it for a single image by beginning the image description with a polygon of the desired color that fills the virtual screen.

Byte #	Description
0	218 (x'DA') - Background color
1	0 - Reserved for future use
2	Background color (0-255); default on power-up = 0

Table 1 Background Color Command Format

CAUTION: *This command must be sent before any object descriptions in the image file.*

(219) RLC Merging Box Data

This command contains RLC image data that is to be merged with a previously specified SCODL image. Refer to Section 2 for a description on *RLC Merging*.

Subcommands 0 and 2 are designated for RLC data (8-bit color codes), which is to be overlaid onto the preceding SCODL data. The RLC data must have 8-bit color codes and they can be either standard 2-byte RLCs (subcommand 0) or the IBM Professional Graphics Adapter (PGA) RLCs (subcommand 2). The SCODL data shows through all areas of the RLC data, which are of the color code specified as transparent in opcode 221 (Transparent Color of RLC Merging Box).

Subcommands 1 and 3 are designated for RLC data with 24-bit color codes. In these cases, the RLC data is reverse merged, indicating that it appears underneath the preceding SCODL data. Since these commands deal with 24-bit color images, the command must be used once for each of the Red, Green, and Blue passes. The procedure for reverse merging a full-color 24-bit image is outlined below. The RLC data can be either standard 2-byte RLCs (subcommand 1) or IBM PGA RLCs (subcommand 3).

The following considerations apply to all subcommands of this opcode:

1. Send the SCODL image first, but leave out the end-of-image code (opcode 128 - x'8000').
2. Specify the dimensions of the RLC data using opcode 220.
3. If any areas of the merged image are not described by either the SCODL data or the RLC data,
   set an appropriate background color using opcode 218.
4. After all SCODL and RLC data has been sent, send the end-of-image code (opcode 128 - x'8000').

Byte #	Description
0	219 (x'DB') - RLC merging box data
1	Subcommand byte 0 - overlays regular 2-byte RLCs; 1 byte for pixel count, 1 byte for color. 1 - underlays regular 2-byte RLCs; 1 byte for pixel count, 1 for single-color intensity. *2 - overlays IBM PGA format RLCs. *3 - underlays IBM PGA format RLCs 5 - underlays regular 2-byte RLC; no expansion
2 to n+1	RLC data; n = number of bytes of data
n+2 to n+3	255 - x'00FF' code, indicates fill to end of image. Use only once.

*Note: The IBM PGA is not currently supported.

Table 1 RLC Merging Box Data Command Format

Specifics of RLC Data

Regular 2-byte RLC data may include the x'0000' code, indicating fill to end of this line. The IBM PGA data is subject to these considerations:

a) The origin (0, 0) is in the lower left-hand corner of the RLC merging box (see opcode 220).

b) There can only be one packet of data per scan line.

c) The image must be described from top to bottom, so the scan line number is always decreasing.

d) There may be gaps in the PGA description at the start or end of each scan line; also, entire scans can be skipped and not described at all.

Procedure

The correct procedure for using subcommands 1 and 3 on 24-bit color images are as follows:

- Opcode 219 must be sent once for each 8-bit color component or three times for a full 24-bit color image. This can be performed by:
- Sending the overlying SCODL image, excluding the end-of-image code
- Sending the Red pass of the image using opcode 219.1 or 219.3
- Sending the Green pass as above
- Sending the Blue pass as above
- Sending the end-of image code

The result is the 8-bit SCODL image on top of the 24-bit color RLC image.

(220) RLC Merging Box Dimensions

This command sets the dimensions of an RLC merging box,-a quantity of image data that is to be merged with another image. Refer to opcode 219 (RLC Merging Box Data) and Section 2 on *RLC Merging* for more detailed information.

The coordinate system, coordinate data format, and the command format are similar to those of opcode 250, which sets the dimensions of a regular output image or *viewport*. These four coordinates determine what part of the underlying SCODL image the RLC box covers.

The default size for the RLC box is in effect when you power up; it is the same as the default output image dimensions.

Byte #	Description
0	220 (x'DC') - RLC merging box dimensions
1	0 - unused
2 - 3	RX1 - leftmost x coordinate of the box
4 - 5	RX2 - rightmost x coordinate
6 - 7	RY1 - top y coordinate
8 - 9	RY2 - bottom y coordinate

Table 1 RLC Box Dimensions Command Format

(221) Transparent Color of RLC Merging Box

This command specifies which color code in the RLC merging box data is to be made transparent and determines where in the box the underlying image is visible.

Refer to Section 2 on *RLC Merging* for a general introduction to this technique.

Byte #	Description
0	221 (x'DD') - Transparent color of RLC merging box
1	0 - reserved for future use
2	transparency color code (0-255); default on power-up = 0

Table 1 Transparent Color of RLC Merging Box Command Format

(222) Macro String Width

This command sets a user-designated width into which an associated string of characters must fit. This width is specified as a SCODL coordinate value between, but not including, the full width of the virtual screen specified by *0* and *2* (x'7FFE' = 1.999939). This command must be used with opcode 234 (Begin macro string), but cannot be used with opcodes 240 (Macro request) or 230 (Catenate macro string).

The character string, whether block spaced or proportionally spaced, is prevented from exceeding this width by reducing the space between each character pair by the necessary amount. Some characters may overlap. This feature also stretches the inter character spacing to fit the specified string width if necessary, but only to a maximum inter character spacing of 2, which represents the width of the virtual screen scaled by the macro scale factor. If this spacing does not stretch the string to the desired width, the string is left-justified and the inter character spacing remains at 2.

Byte #	Description
0	222 (x'DE') - Macro string width
1 - 2	String width: 0 = no justification. Negative values are ignored. The binary point is to the right of the 2nd digit.

Table 1 Macro String Width Command Format

This command is designed to be used for *text justification*. Justification is the process of stretching or compressing strings of characters so that the left and right margins of a body of text will be uniformly aligned. You may assign the same macro string width to a number of vertically stacked strings of characters whose base addresses all have the same X coordinate. This causes the text to create a flush right-hand margin, except for strings such as the last string in a paragraph that are too short. These remain left-justified only. The suggested procedure for justifying a block of text is as follows:

1. Calculate the desired width (x) of the text on the virtual screen.
2. Calculate the approximate total length of your text or use opcode 130 (Return Macro String Length). If it is to be proportionally spaced, add the character widths for the internal character set provided in Table 3-3 in Section 3, *Describing Macros and Text*.

   If it is to be block spaced, add the block widths at 2.0 per character plus all the inter character spaces. Remember to apply your macro scale factor to the result.
3. Divide the text into segments of calculated length (x).
4. In the SCODL file, send commands in this order:

- (233) String mode and spacing; (242) macro colors and (244) scale factor;
- (222) macro string width (x);
- (243) base address (a) of first line of text;
- (234) first line of text;
- base address (b) of second line of text;
- second line of text;
- continue adding lines in this way until the end of text.

NOTE: *It is recommended that the last line in each paragraph be assigned a string width of* **0** *so that it will be spaced according to the specified string mode and spacing, with no attempt to justify.*

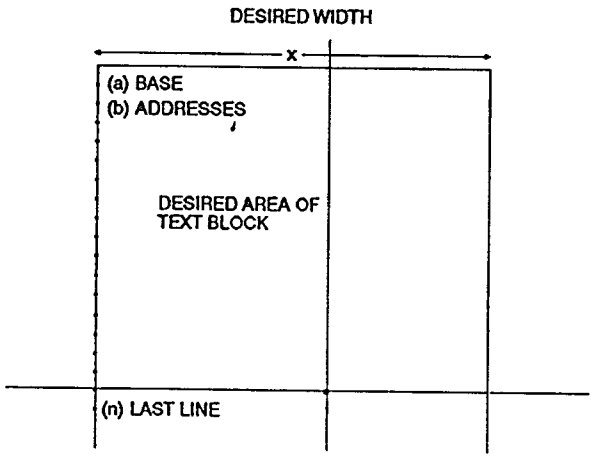


Figure 1 Justifying a Block of Text

(223) Select Output Device

This command enables you to switch output devices during normal MVP+ processing without requiring a power up or reset. All images received after this command are generated according to the requirements of the selected output device. Any object descriptors preceding this command are thrown out when the command is received. The default output device is the null device.

Byte #	Description
0	223 (x'DF') - Select output device
1	Output device code number 1 = null device (default) 2 = QCR or PCR film recorder 3 = Diablo ink-jet printer 4 = Tektronix 4695 ink-jet printer 5 = Seiko thermal transfer copier 6 = Hewlett Packard Laser Jet Series II 7 = Mitsubishi G650 thermal printer * 8 = Tecmar screen-display system 10 = IBM Enhanced Graphics Adapter (EGA) * 11 = IBM Professional Graphics Adapter (PGA) 12 = Matrix TT200 thermal printer 13 = IBM Colorjet ink-jet printer * 14 = IBM Proprinter 15 = Tektronix 4692 ink-jet printer 16 = Tektronix 4692 transparencies 18 = Canon LBP-8 A2 laser printer 19 = Lasercomp 20 = Targa 16, 24, 32 graphics adapter 21 = NNGS Revolution graphics adapter 25 = Calcomp Plotmaster thermal printer
2	Output device mode For devices 1, 3, 4, 5, 8, 12, 16, 18, 19, 20, 21: set to 0 - no other mode available For device 2 (QCR or PCR), set bit 0 as follows: 0 = 2K resolution mode 1 = 4K resolution mode For devices 10 and 11 (EGA and *PGA), set bit 0: 0 = 560 horizontal pixels 1 = 640 horizontal pixels

* not available in this release.

Table 1 Select Output Device Command Format

(224) SCODL Control Command

This opcode contains subcommands that alter certain parameters of SCODL processing. It must be sent before any object descriptors in the image. Any objects preceding this command are ignored when this command is received.

Byte #	Description
0	224 (x'E0') - SCODL control command
1	1 - Subcommand 1 - reserved
	2 - Subcommand 2 - reserved
	3 - Subcommand 3 - Reset SCODL table sizes
	4 - Subcommand 4 - Reset macro structure
	5 - Subcommand 5 - Turn on virtual screen clipping mode
	6 - Subcommand 6 - Turn off virtual screen clipping mode
	10 - Subcommand 10 - Select SCODL coordinate scaling

Table 1 SCODL Control Command Format

Subcommand 3

Resets the SCODL table sizes to their default values. Refer to opcode 239.

Subcommand 4

Resets entire macro structure. All user-defined macros and patterns are erased.

Subcommand 5 & 6

This command is used when SCODL object points lie off the virtual screen. This can happen if macros and thickened lines extend beyond the virtual screen dimensions. The error message **?39 FNFN** is displayed to warn you that image distortion may exist.

Subcommand 5 (turn clipping on) may be used to eliminate this distortion, but reduces the size of the SCODL input coordinates relative to the virtual screen by a factor of two in both X and Y dimensions. Also, more memory is required and drawing precision is reduced by one bit, from a fourteen bit fraction to thirteen. The precision loss is negligible for output images less than or equal to 16K. Above 16K, clipping has no effect.

Subcommand 5 must only be sent at the beginning of an image or before the first object descriptor, such as a line or arc. Object descriptors preceding this command do not appear in the final image. Clipping mode stays on until superseded by subcommand 6.

Subcommand 10

Allows binary shifting of incoming SCODL coordinates. This function is useful if an application program has created SCODL files that do not use the entire virtual screen and needs to be scaled up for higher output resolution.

The format for the command is

Byte #	Description
0	224 (x'E0') - SCODL Control Coommand
1	10 - Select SCODL Coordinate scaling
2	nn - Number of shift lefts (multiply by 2 to the power of nn, where 8 <= nn <= 8). A negative nn means shift coordinates to the right (divide by 2 to the power of nn). If nn=0, no scaling is done (default on power up).

Table 2 Subcommand 10 Command Format

This command may be used to make all SCODL objects 2, 4, 8, 16, and so on, times larger or smaller in both X and Y directions. This is a specialized command that can only be used if it is known that the input SCODL file describes all its objects in a centered portion of the virtual screen. If a SCODL image does not fill the virtual screen, then the maximum resolution that the image can be output is reduced.

For example, suppose a SCODL file had all its objects described in a centered box 1/16th the size of the virtual screen (1/4 used in each direction). Since the maximum output dimension of the virtual screen is only 8192 pixels in X and Y. If we send the select SCODL coordinate scaling command with a shift count of 2 (multiply by 4) before this SCODL image, the image fills the virtual screen and the maximum virtual screen resolution may be output.

This command should not be used to enlarge objects past the edge of the virtual screen as the results will be undefined. This command has no effect on RLC merge commands (opcode 219) as this data is fixed resolution data and can not be sized in any way.

The command must be sent before any object descriptors in the image. Any objects preceding this command are ignored when this command is received. The SCODL coordinate scaling selected stays in effect until superseded by another subcommand 10.

> ***NOTE:*** *All thickening (opcode 216) is turned off when this command is received. Thickening can be turned back on after this command. This feature only has an effect if thickening from the previous image also applies to this image.*

(225) Load Selected LUT Values

This command allows you to load some of the values in a look-up table without loading the entire LUT by using one of opcodes 251 to 254.

The addresses is filled in ascending order starting with byte #2 as outlined in Table 1. If the number of bytes being sent is greater than the number of LUT addresses following the first LUT location to be loaded, the new values wrap around to the beginning of that LUT. A single use of this command can only modify the contents of one LUT.

Byte #	Description
0	225 (x'E1') - Load selected LUT values
1	Destination LUT: 0 = Neutral 1 = Red 2 = Green 3 = Blue
2	First LUT location being loaded (0-255)
3	No. of bytes (N) being sent (0-255); if 0 is specified, 256 bytes are transferred.
4 - N+3	New LUT values

Table 1 Load Selected LUT Values Command Format

For example, to reset locations 4, 5, and 6 in the red LUT with intensity values (125, 80, 72), the following string is used:

(Dec.) 225 1 4 3 125 80 72
(Hex.) x'E1 01 04 03 7D 50 48'

(226) Set Virtual Screen Image Rotation

This command sets the MVP+ in a mode in which all geometric objects with SCODL coordinates are rotated around the virtual screen center by a number of degrees specified by a subcommand byte. The command must appear before any object descriptors in the image to be rotated. Parts of the image that are not objects, such as the output image window or *viewport* - the portion of the virtual screen to be output, are not rotated.

The rotation mode selected remains in effect until replaced by another subcommand, that is, another use of opcode 226. The default when you power up is no rotation, or subcommand 0.

Byte #	Description
0	226 (x'E2') - Set Virtual Screen Image Rotation
1	Rotation mode: = 0 - rotate 0 ° = 1 - rotate 90 ° counter-clockwise = 2 - rotate 180 ° counter-clockwise = 3 - rotate 270 ° counter-clockwise

Table 1 Set Virtual Screen Image Rotation Command Format

For example, if the string **226 2** (x'E2 02') precedes the object descriptions represented in Figure 1 (a), the objects appear in the image as they do in (b). Note that the output viewport has not moved.

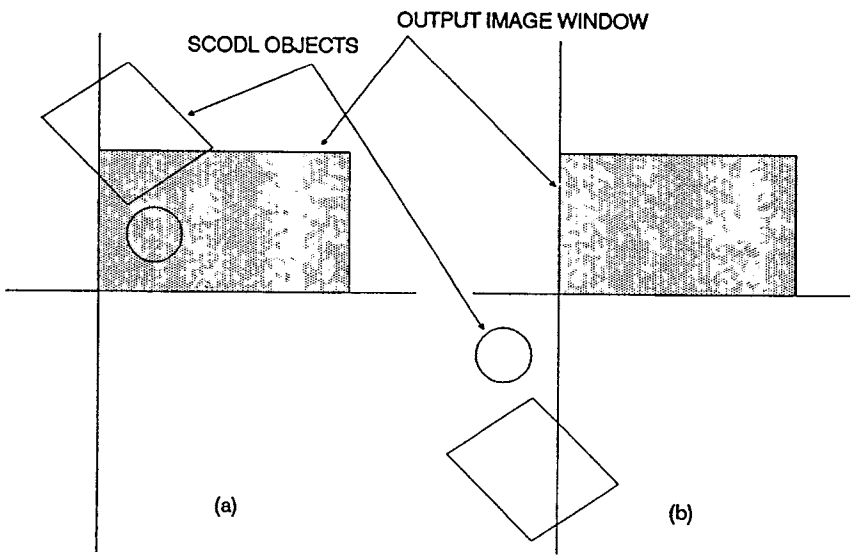


Figure 1 Example of Virtual Screen Image Rotation

(227) Define a Macro

This command allows you to define macro object descriptions such as those for characters and to class these macros into families, such as sets of characters of a given style or font. Any macro that is defined using this command is given the last set of kerning points specified via opcode 247 (Define Kerning Points). Kerning points allow macros defined in this way to be proportionally spaced via opcode 233.

Macros must be defined before the object descriptors in an image. They remain in memory from one image to the next. Macro definitions can not be nested. That is, one macro definition can not be part of another.

Macro storage requires memory space that is taken from general system memory. Storage of a large number of macros can affect the performance of the system and the number that can be stored at once is limited. Defined macros may be erased by redefining them with no data bytes between the macro code and opcode 241 (End of Macro Definition).

All data entered between opcode 227 (x'E3') and 241 (x'F1') is stored as a single macro and processed as SCODL commands. However, the extent to which SCODL commands can be nested within a macro definition is limited and the MVP+ does not contain a means of checking for unworkable commands. You are advised to use caution when building macros out of opcodes other than object descriptors.

A macro definition can not extend beyond the edges of the virtual screen, so be careful not to include arcs, which may do so. Also, macro definitions can not contain an end-of-image command (opcode 128).

Byte #	Description
0	227 (x'E3') - Define a macro
1	Macro family code (1-127)
2	Macro code (1-254)

Table 1 Define a Macro Command Format

For example, to define the uppercase *L* of macro family code 4 in colors 23 and 56 as illustrated in Figure 1, the following string is used:

(Dec.) 227 4 76. 4 23 0 56 -32766 32766 (other vertices) -32766 -32767. 241.
(Hex.) x'E3 04 4C 04 17 00 38 FF FE 7F FE (other vertices) FF FE FF FE F1

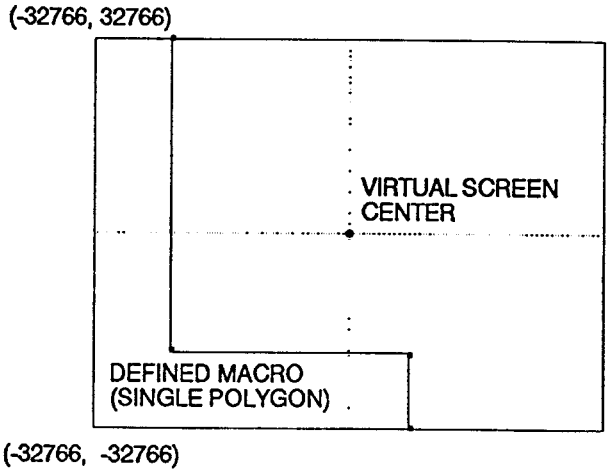


Figure 1 Example of a Macro Definition

(230) Catenate Macro String

This command is identical to Opcode 234 (Begin a Macro String), except that it is used when another string is already present to which the string following Opcode 230 can be catenated or tagged on. The new string follows the last character of the earlier string. It is not included in any right, left, or center justification, which was performed on the earlier string.

The purpose of this command is to free you from the need to restate the string mode and spacing. However, you have the option of specifying a new string mode and spacing, if desired, and still have the new string catenated behind the old one. When catenating a string, remember to include a space unless the break between strings occurs in the middle of a word.

Byte #	Description
0	230 (x'E6') - Catenate macro string
1	1st macro request
2	2nd macro request
.	.
n	last macro request; n = total number of macros in string
n+1	255 (x'FF' - end-of-string character)

Table 1 Catenate Macro String Command Format

Macro strings actively adjust the base address as characters are generated. When the whole string has been generated, the base address is to the right of the last character. Keep in mind that when the original and catenated strings are separated by other data, that data may affect the location of the base address.

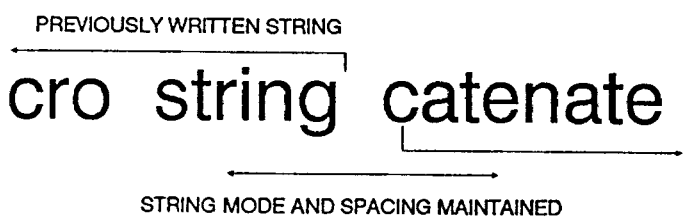


Figure 1 Example of Catenation of Macro Strings

(231) Select String Font

This command selects the character font called a *typeface* to be used in subsequent macro string commands (opcode 234). This is done using the macro family code attached to macros. If font zero is selected, or if a string is requested without a font having been specified, macros are taken from SCODL's internal character set. Individual characters of any font can also be requested through opcode 240 (Macro Request). Because it allows a byte for the family/font code, it does not require an accompanying opcode 231.

When a font is selected, it is loaded into memory. The memory space occupied in this way can be freed again using opcode 237 (Delete macro family/font). The format for this command is illustrated in Table 1 below.

Byte #	Description
0	231 (x'E7') - Specify String Font command
1	Desired font 1-127 (default = 0)

Table 1 Specify String Character Font Command Format

(232) Macro Rotation

This command allows macros or strings of macros to be rotated relative to the virtual screen. The angle of rotation is measured counter-clockwise. Single macros are rotated around the current base address. Strings are rotated around their justification points, whether left, right, or center as illustrated in Figure 1.

The sine and cosine of the angle of rotation are positive or negative values between 1 and -1, stated in the same format as SCODL coordinate values. The rotation is performed on all macros following this command until a new rotation command is received. Rotation can be stopped by specifying 0 for both the sine and cosine of the angle of rotation.

Byte #	Description
0	232 (x'E8') - Macro Rotation command
1-2	Cosine of angle of rotation
3-4	Sine of angle of rotation

Table 1 Macro Rotation Command Format

For example, if the strings in Figure 1 are rotated counter-clockwise by 90 degrees (sine = 1; cosine = 0), the following command string is used:

(Dec.) 232 0 32766 (Hex.) x'E8 00 00 7F FE'

This command gives the rotated strings represented in Figure 1. The strings rotate around their base addresses. The default setting for the macro rotation is zero degrees. The MVP+ returns to this default between images.

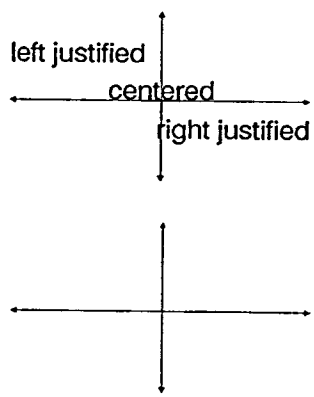


Figure 1 Examples of Macro String Rotation

(233) Specify String Mode and Spacing

The location of a macro on the virtual screen is primarily determined by the base address set by opcode 243. However, this does not determine what part of the macro is placed over the base address. Furthermore, in strings of text there must be a system for determining how each successive character is placed relative to the first.

Since characters have varying widths and shapes, good typography requires a fairly complex formula for positioning adjacent characters. Some of the factors affecting macro string placement are specified by this opcode, the format for which is given in Table 1 below.

Note that since the default spacing mode is 0, no defaults exist for bytes 2 to 4.

Byte #	Description
0	233 (x'E9') - String mode & spacing.
1	Spacing mode 0 - No automatic inter character spacing (default) 1 - Left justified, block mode spacing 2 - Left justified, proportional mode spacing 4 - Centered, proportional mode spacing 5 - Centered, block mode spacing 8 - Right justified, proportional mode spacing 9 - Right justified, block mode spacing 18 - Left justified, kerned spacing 20 - Centered, kerned spacing 24 - Right justified, kerned spacing 50 - Left justified, kerned spacing, with kerning table 52 - Centered, kerned spacing, with kerning table 54 - Right justified, kerned spacing, with kerning table
2	TXA - Text Alignment. Specifies the relationship between the first character of a string and the starting base address. Figure 1 illustrates the 35 different starting points. 0 TXA 34
3 - 4	ICS - Inter Character Spacing. Specifies the amount of additional space to be left between characters. ICS is a signed fraction in SCODL format with a range of -2.0 ICS < +2.0. 2.0 is a character block's width. The maximum value of x'7FFF' leaves this width between characters or blocks. There are 2 bits before the binary point in the 2's complement 16-bit fraction. The ICS value is subject to the macro scale factor (opcode 244).

Table 1 String Mode and Spacing Command Format

Text Alignment Points

The problem of aligning a macro precisely with the base address is solved with text alignment points. Each of SCODL's internal character descriptions is horizontally centered in the virtual screen and is surrounded by thirty-five points; these locations are illustrated in Figure 1. Byte # 2 of this opcode allows any one of these points to be selected as the one that coincides with the base address.

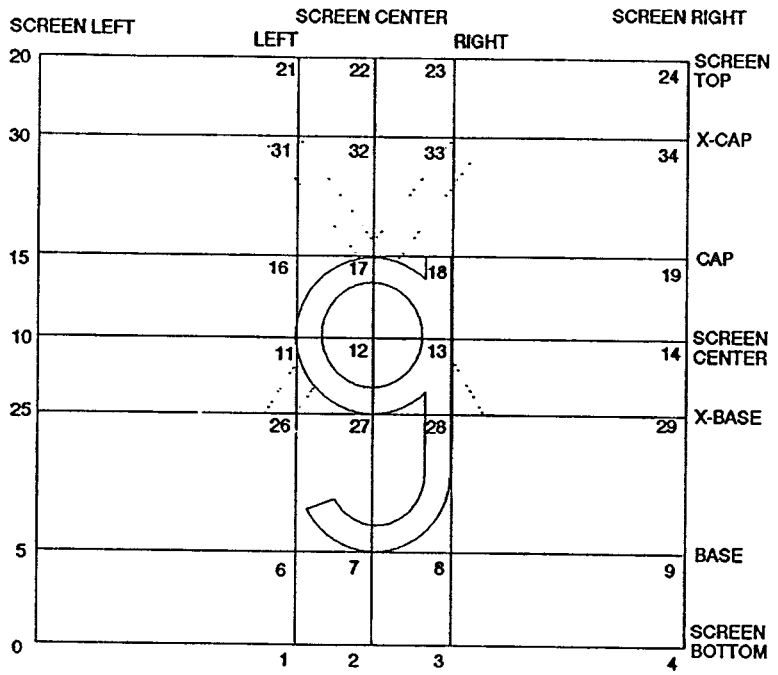


Figure 1 Character and Text Alignment Points on the Virtual Screen

The rows of text alignment points from 25-29 and 30-34 (X-base and X-cap respectively) are consistent for all of SCODL's internal characters. Using the upper case X as a basis, all characters in the set have been defined so as to align typographically on these lines.

For user-defined character sets X-base, X-cap, and the vertical locations of their text alignment points are set by opcode 217. On the other hand, the lines that define the boundaries and center lines of the virtual screen defined in Figure 1 as screen left, screen right, screen top, screen bottom, and the two screen center lines, are arbitrary and varying distances from the edges of the character. They would allow perfect base address alignment of any macro that touched the edges of the virtual screen.

A similar function is performed by the left, right, cap, base lines that define the extreme horizontal and vertical edges of the character without reference to the rest of the character set. Their intersections illustrated in Figure 1 as text alignment points 6, 8, 16, and 18, define the minimum rectangle that encloses the character. These allow the character or other macro to be aligned precisely with the location of the base address and therefore with any corner, edge, or point on the virtual screen.

Spacing Modes

When macros are being requested in a string, the base address and text alignment point are only specified once. One character is positioned and the others are laid down relative to it according to rules you set. Provided the chosen text alignment point is on x-base or x-cap illustrated in Figure 1, SCODL's internal character set follows a correct typographic baseline. However, the horizontal spacing between them is governed by the your choice of a spacing mode.

You can select whether to give all characters uniform amounts of screen space based on the virtual screen called *block mode* or to give them space in proportion to their actual widths called *proportional mode*. Proportional spacing is more attractive and legible while block spacing has the advantage of allowing you to calculate easily in advance how much space a string occupies.

Kerned Spacing

Kerned spacing is similar to proportional mode, but goes to greater lengths to produce optimal spacing between pairs of characters based on their shapes. Kerned spacing with kerning tables enables the use of a kerning table if one has been supplied for this font (see opcode 214).

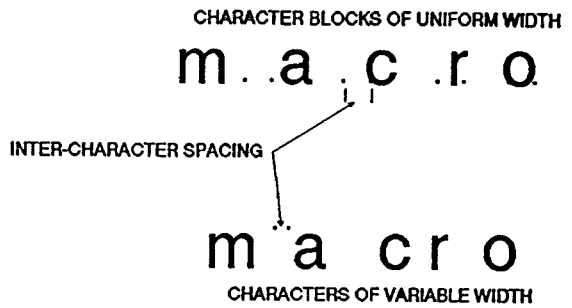


Figure 2 Block Mode and Proportional Mode Spacing

Character Alignment

The descriptions of macro string width (opcode 222) and other string commands have implied that the first character requested is the one aligned with the base address. The base address moves along the string as each character is written in an amount determined by the width of the character (block or proportional) and the inter character spacing. You can even cause a string to be written from right to left by specifying a fairly large negative value for inter character spacing.

Aligning the starting character with the initial base address is not the only option for string alignment. Strings of macro characters may be centered or may have their last rightmost characters aligned with the base address, making it possible to present a flush right margin in a body of text. Hence, the spacing mode byte in opcode 233 encodes a choice not only of block or proportional spacing, but of left, right, or center justification.

When a string is being centered, only text alignment points on the vertical center line are valid. These points are numbered 2, 7, 12, 17, 22, 27, and 32 in Figure 1. Use of any other point gives undefined results. The string's horizontal placement is such that the base address is at its measured center point. When a string is being right justified, the text alignment point is understood to apply to the last character.

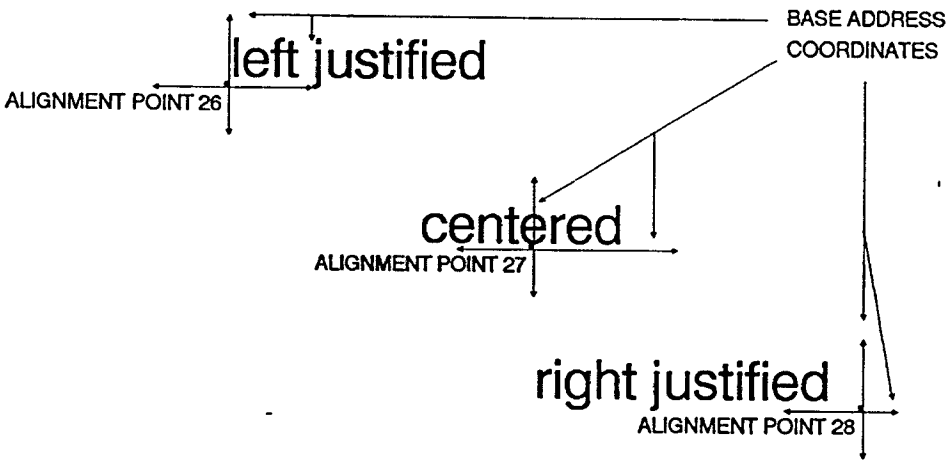


Figure 3 Left, Right and Center Justification

Inter Character Spacing

Inter character spacing (ICS) is a the amount of blank space you select to be inserted between all adjacent characters in a string. The ICS byte in opcode 233 specifies this value as a fraction of the virtual screen width of 2, but it is subject to the same scaling factor set for the macros.

In block mode strings, the ICS is simply inserted between the uniform character blocks, even though these blocks generally already contain white space on either side of the character. Each character block is simply the virtual screen on which the character was defined and scaled to the same factor as the macro.

In proportional mode and kerned mode, the objective is to produce typographically pleasing, easily legible strings of text. To this end, it is sometimes not enough to measure the ICS from the absolute horizontal left and right extremes of the characters, as is done in proportional mode. With such a simple technique, character pairs such as uppercase **A** and **V** would be placed too far apart to satisfy the demands of good typography.

This problem is resolved by the kerned spacing modes. These modes define the left and right edges of a character using six kerning points three per side; the spacing between adjacent characters is measured from the pair of shared kerning points - whether top, middle, or bottom - closest to each other. As you can see in Figure 4, this can allow the absolute horizontal limits of the characters to overlap in cases where their shapes make wider spacing inappropriate.

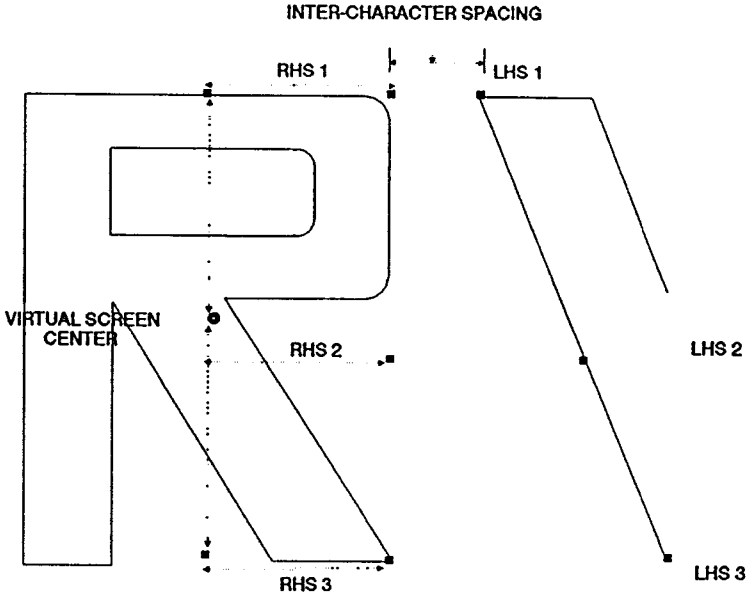


Figure 4 Inter Character Spacing and Kerning Points

For more information on kerning points, refer to the description of opcode 247.

Example: The command string **233 4 27 0.4** applied to the string **macro** selects proportional but unkerned spacing, producing the string represented by Figure 5:

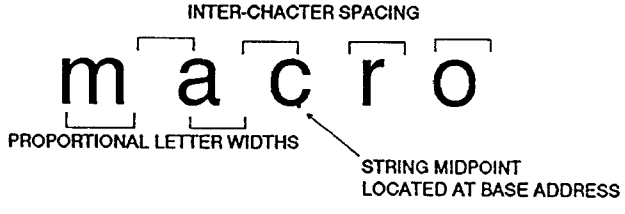


Figure 5 Example of the use of the Specify String Mode and Spacing Command

Replacing the second byte in the command string (4) with a 20 would select kerned spacing. The character string is again centered, but the effect of kerning is to draw the **ac** and **ro** character pairs slightly closer together.

Setting the same byte to 52 enables the use of the kerning table, if one had been supplied using opcode 214. A kerning table lists character pairs, which are exceptions to the normal kerning system, and gives combined width values that overrule regular kerning for those pairs.

The table of SCODL internal macros listed in Appendix A, provides kerning points for all characters in the internal set. This table allows the user to predict string lengths exactly as necessary.

The string mode and spacing specified by opcode 233 may be subject to adjustment by opcode 222, which specifies a desired width for a macro string. Opcode 222 also makes it possible to justify a block of text, such that it creates flush left and right margins when performing an output. Refer to the description of this opcode for more information.

(234) Begin a Macro String

Bytes following the 234 opcode (x'EA') are interpreted as Macro requests until an end-of-string character (x'FF') is received. Refer to the list of SCODL internal macros in Appendix A. The macros in the string are assumed to belong to the family/font specified by Opcode 231 and are therefore only identified by their macro codes. Character size is specified by the Macro Scale (opcode 244), and inter macro spacing is specified by opcode 233.

Macro strings actively adjust the base address as characters are generated. When the whole string has been generated, the base address is to the right of the last character.

Byte #	Description
0	234 (x'EA') - Begin macro string opcode
1	1st macro request
2	2nd macro request
.	.
.	.
n	last macro request; n = total number of macros in string.
n+1	255 (x'FF' - end-of-string character)

Table 1 Begin Macro String Command Format

For example, to define the macro string **macro** in the internal character font (macro family 0), centered, and proportionally spaced with X and Y scaling factors of 0.1 and 0.1, the following string is used:

(Dec.) 231 0 . 233 4 7 1638 . 244 3277 3277 . 234 (macro codes) 255 .
(Hex.) x'E7 00 E9 04 07 06 66 F4 0C CD 0C CD EA (macro codes) FF'

Note that the scaling factors are encoded in the same format as coordinate values and therefore must be multiplied by 32768; the number of positive virtual screen addresses:

0.10 x 32768 = 3277 (x'0CCD')

(235) Specify Compressed Mode Weighting Function

This command is used along with Opcode 8 (Compressed Straight Edge polygon) to describe compressed polygons. The byte assignment for this command is given in Table 1 below. This command specifies the function that is to be applied to each differential coordinate when mapping the coordinate into the virtual screen. Each differential coordinate is multiplied by 2^W prior to being added to the previous vertex coordinate.

Byte #	Description
0	Opcode 235 (x'EB')
1	reserved
2	weighting function 0 w 8

Table 1 Compressed Mode Weighting Function Command Format

For example, to draw a compressed mode polygon in the size its coordinates were originally defined, the polygon description is preceded by the following command string:

235 0 0 (x'EB 00 00')

Refer to the discussion of Opcode 8 for more information on the use of this command. Note that the weighting function only increases the distances between the coordinates; a weighting function of 0 leaves these distances as they were defined. A compressed mode polygon must therefore be defined in the smallest size the user intends to use it in an image.

The default setting for the weighting function is zero; the MVP+ returns to this default between images.

(236) Output Device Dependent Command

This command has different meanings, depending on the output device being used. It specifies the position of the output image on the output device screen, or working area, and in the case of ink-jet printers, it also allows specification of the dot matrix size.

Using a QCR Film Recorder

When the output device is a film recorder, this command allows the user to place the image described by the current SCODL file anywhere on the film recorder's output area. You must specify the coordinates of the upper leftmost pixel called the *start pixel* of the SCODL output image, in terms of pixels of the film recorder.

The start pixel coordinates tell the film recorder where to start drawing the rectangular array of pixels defined by the output image. The output image must fit within the amount of film area that is to the right of and below the start pixel's screen position.

Byte #	Description
0	236 (x'EC') - Output device dependent command
1	1 - specifies the start pixel
2 - 3	QCRSTX - start pixel x-coordinate on film recorder
4 - 5	QCRSTY - start pixel y-coordinate on film recorder

Table 1 Specify Film Recorder Start Pixel Command Format.

Examples:

For normal full-screen 2K QCR operation using the default mode when you power up, the start pixel coordinates used are (-1024, +767) or (x'FC00', x'02FF'). The start pixel command for 2K is:

236 1 -1024 767 (x'EC 01 FC 00 02 FF')

The start pixel coordinates for full-screen 4K QCR operation are (-2048, +1535) or (x'F800', x'05FF'). The start pixel command for 4K is:

236 1 -2048 1535 (x'EC 01 F8 00 05 FF')

When the output device is an ink-jet printer, a thermal transfer copier, or a video graphics display, two subcommands of opcode 236 may be used.

Subcommand 1

This subcommand starts pixel X and Y coordinates

This subcommand specifies the position of the image's start pixel relative to the top left pixel in the output area, rather than to a central point of origin. The X and Y coordinates are stated as a number of output pixels and are always positive. Negative values are forced to 0.

For example, if x = x'0010' and y = x'0020', the output image is 16 (x'10') pixels to the right of the left margin and 32 (x'20') lines are skipped before the start of the image.

Byte #	Description
0	236 (x'EC') - Output device dependent command
1	1 - Specify Start Pixel Coordinates
2 - 3	Start pixel x coordinate
4 - 5	Start pixel y coordinate

Table 2 Non-QCR Start Pixel Coordinates Command Format

Subcommand 2

This subcommand specifies dot matrix size

This subcommand specifies the size of the dot matrix within which the eight basic colors are combined to produce further shades of color. The larger the dot matrix, the larger the number of colors that can be used. This subcommand specifies the dot matrix size as 1, 2, 4, or 8. For example, 4 specifies a 4 x 4 dot matrix.

The MVP+ employs dot matrices in ways that maximize geometric resolution while maintaining color choice. However, setup time increases with large matrix sizes. The recommended dot matrix size for the best possible color resolution is 8 (64 dots), which gives 65^3 possible shades of color. Only the 256 colors coded into the look-up tables can be selected for any one image.

Byte #	Description
0	236 (x'EC') - Output device dependent command
1	2 - Specify dot matrix size
2	Matrix size in dots, each direction (1, 2, 4 and 8)
3	0 - Unused

Table 3 Specify Dot Matrix Size Command Format

For example, consider the creation of a scale of grey shades between white and black using only pixels of those two colors illustrated in Figure 1. If a dot matrix size of 2 is selected, three shades of grey can be created between all white (matrix 0) and all black (matrix 4). The same method is used to create intermediate tones of any one of the basic colors or with combinations of pixels of more than one color.

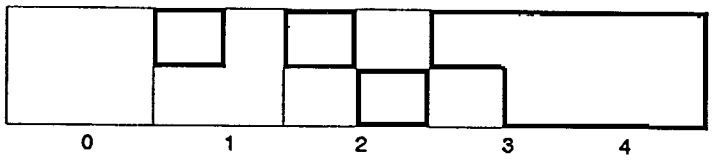


Figure 1 Possible Grey Shades (dot matrix size = 0)

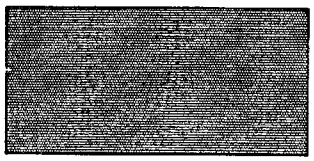


Figure 2 Grey Shade Created Using #3 Matrix from Figure 1

(237) Delete Macro Family/Font

Macros are grouped into families or fonts called typefaces. This command deletes an entire family or font from memory, freeing the memory space that was taken when that family/font was selected via opcode 231 (Select family/font).

Byte #	Description
0	237 (x'ED') - delete Macro Family/Font
1	0 - unused
2	Family/font code (0 = MVP+ internal font)

Table 1 Delete Macro Family/Font Command Format

(238) Specify Frame Reference Number

The SCODL processor assigns an increasing frame reference number to each image it generates. This frame reference number is reported back to the host processor when any errors are detected by the SCODL processor. Normally, the initial value for the reference number is one. This command allows the frame reference number starting value to be initialized to some number other than one. The frame reference number may be issued at any point in the SCODL data stream; however, it is suggested that it be sent at the beginning of an image before any descriptor codes. The format for this command is found in Table 1 below.

CAUTION: *It is recommended that this command* **not** *be used for an image that is to be repeated, that is, one with an image repeat (opcode 245) of greater than 1. Otherwise, confusion in frame counting may result.*

Byte #	Description
0	238 (x'EE') - Specify frame reference number
1 - 2	FRMRNO - Two-byte value specifying frame reference number value

Table 1 Frame Reference Number Command Format

(239) Specify SCODL Table Sizes

This command modifies the amount of memory allocated to a set of tables within the SCODL program. Before running an image, SCODL reserves a fixed memory area for various tables. These tables have been set up to accommodate the processing of most imagery, however, under certain circumstances, an image may overflow one of the tables.

When this happens, an error message is output, and the current image is aborted. Table sizes will have to be modified to run this image. The format for this command is provided in Table 1 below. The values set by this command remain in effect until superseded by a subsequent opcode 239 command or until a reset.

This command must only be sent at the beginning of an image before the first geometric object descriptor. Object descriptions that precede this command do not appear in the final image.

Byte #	Description
0	239 (x'EF') - Specify SCODL table sizes
1 - 2	NUMOBJ - Number of Objects NUMOBJ must be greater than or equal to the total number of objects in the SCODL description. Each edge, non-filled arc, or polygon SCODL command counts as one object. A filled arc is counted as two objects. When macros are used, each object within the macro must be counted. 0 < NUMOBJ < 65532
3 - 4	NWRKSLT - Number of Work Slots Maximum number of overlapping objects per scan line. This number is a measure of image complexity. This table is used in the hidden surface removal algorithm. 0 < NWRKSLT < 32767
5 - 6	NCHKSLT - Number of Check Slots Maximum number of edges that are covered by an object on a given scan line.This number is a measure of image complexity. 0 < NCHKSLT < 32767
7 - 8	NPRIO - Maximum Number of Priority Elements This is the maximum number of filled objects that can exist on any scan line. 0 < NPRIO < 32767
9 - 10	NRLC - Maximum Number of RLCs This is the maximum number of RLCs needed to describe any scan line. 0 < NRLC < 32767

Table 1 Specify SCODL Table Sizes Command Format

SCODL Table Default Sizes

Upon starting up the MVP+, the SCODL table sizes are given default values. Table 2 below lists these default values. These default values work for images up to a particular level of complexity.

The table sizes may need to be changed, however, as image complexity increases. The sizes specified should be kept as small as possible to reduce memory usage, yet they should remain large enough to handle the current set of images.

Field	Default Value On Power Up
NUMOBJ	8000
NWRKSLT	100
NCHKSLT	500
NPRIO	100
NRLC	1024

Table 2 SCODL Table Sizes - Default Values (decimal).

(240) Macro Request

This command requests a single macro, using both its family/font code and its macro code. The family/font code selected here do not affect subsequent macros.

The macro is colored according to the Macro Color command (opcode 242) and is sized according to the Macro Scale command (opcode 244). It is positioned with text alignment point 12, the center of the virtual screen on which it was defined, positioned over the base address.

Byte #	Description
0	240 (x'F0') - Macro request
1 - 2	Macro Request Code - Specifies the requested macro's family/font (0-127) and macro (1-254) code number. Calculated using the following formula: *(Font code x 256) + (Macro code)* = 2-byte request code The 2-byte code gives you the option of having up to 256 2 macros without classifying them by font.

Table 1 Macro Request Command Format

> ***NOTE:*** *This command only allows you to request one macro each time it is used. To request a string of macros, use opcode 234 (Begin a Macro String), preceded by opcode 231 (Specify String Character Font) and ending the string with the end-of-string character 255 or x'FF'.*

The macro code in opcode 240 is two bytes long to include the macro family (character font) code, making opcode 231 unnecessary with this command. If you are in a string spacing mode other than 0 (see opcode 233), and you attempt to request an individual macro using opcode 240, the results will be undefined. Opcode 233 should, if used, be reset to spacing mode 0 to request a single macro.

(241) End Of Macro Definition

This code is only legal when defining a macro; if used otherwise, an error results. It specifies the end of a definition of a macro via opcode 227.

Byte #	Description
0	241 (x'F1') - End of Macro Definition

Table 1 End of Macro Definition Command Format

(242) Specify Macro Colors

This command specifies the colors to be applied to macros: The default settings for both colors are zero. The MVP+ returns to this default between images.

Objects in macro definitions may contain color specifications. These colors are replaced by those specified in opcode 242 or by its default colors, except if subcommand 1 of opcode 210 (Macro color control) is used to enable the colors contained in the original macro definition.

Byte #	Description
0	242 (x'F2') - Specify macro colors
1	HC - highlight color
2	FC - fill color

Table 1 Specify Macro Colors Command Format

(243) Specify Macro Base Address

This command specifies the base address at which requested macros are to be located. A base address can be positioned anywhere within the virtual screen area, provided the macro is scaled and positioned, and does not extend beyond the limits of the virtual screen. Macros extending beyond the virtual screen boundaries give undefined results.

Strings of macros are positioned relative to the base address determined by the text alignment point given by opcode 233 (Specify String Mode and Spacing). Single macros requested by Opcode 240 (Macro Request) have their text alignment point #12 at the center of the virtual screen on which they are defined coincident with the base address.

Byte #	Description
0	243 (x'F3) - Specify macro base address
1 - 2	XBASE - X Coordinate of the Base Address defined in standard coordinate format
3 - 4	YBASE - Y Coordinate of the Base Address defined in standard coordinate format

Table 1 Specify Macro Base Address Command Format

The default setting for the Macro Base Address coordinates is zero or the center of the virtual screen. The MVP+ returns to this default between images.

(244) Specify Macro Scale

This command specifies the scaling function to be applied to all object descriptor coordinates in macros when they are applied to an image. As originally defined, macros take up most of the virtual screen. Thus, macros can not be expanded with the scale factor command.

The scale factor is treated as a proper fraction, that is, a number less than one. It is in standard coordinate format, thus allowing negative scales that create mirror images of macros.

Byte #	Description
0	244 (x'F4') - Specify macro scale
1 - 2	XSCALE - X Coordinate scale factors in standard coordinate format
3 - 4	YSCALE - Y Coordinate scale factors in standard coordinate format

Table 1 Specify Macro Scale Command Format

The default setting for the macro scaling factor is zero or *no scale*, in which case the macro occupies as much of the virtual screen as it did when originally defined. MVP+ returns to this default between images.

Arcs may appear distorted after asymmetric scaling. When scaling internal macro characters, you are advised to keep the absolute values of XSCALE and YSCALE equal.

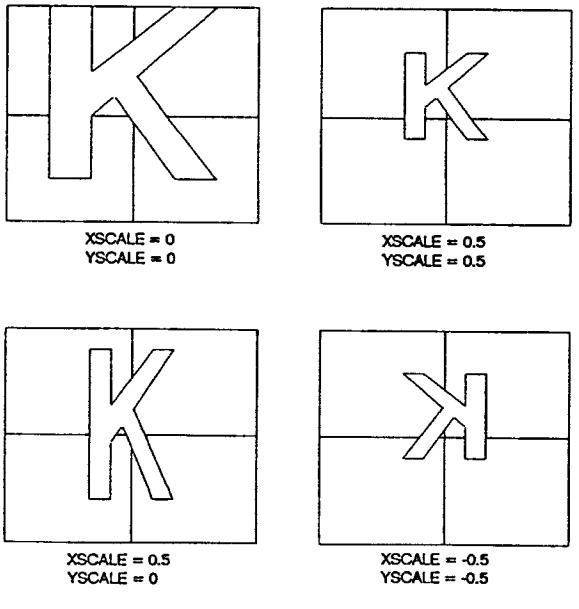


Figure 1 Various Uses of Macro Scaling

(245) Image Repeat Request

This command requests that the following SCODL image description be imaged the specified number of times. The format for this command is found in Table 1 below.

Byte #	Description
0	245 (x'F5') - Image repeat request
1	REPEAT - Single byte specifying the number of times that the image is to be repeated. A repeat of zero is interpreted as a count of 256. 0 <= REPEAT < 255

Table 1 Image Repeat Request Command Format

The default setting for the image repeat is one image; the MVP+ returns to this default between images.

(246) GPIB Data Pass Through

This command is used to pass data directly between the host processor and devices on the GPIB interface bus. For instance, it can be used to send direct instructions to a QCR digital film recorder to disable the camera module's film wind. This permits several images to be exposed on a single piece of film.

Constraints

This command is intended primarily to send to and receive from the QCR film recorder. Data can be written to various GPIB devices but can only be read from a QCR. Only primary GPIB addressing and serial poll interrupts are supported. All data returned to the MVP+ is in ASCII-encoded hexadecimal format.

Position in File

This command must only be sent at the beginning of an image before the first geometric object descriptor. Object descriptions that precede this command do not appear in the final image. When this command is used with image files larger than 64K, you must send image data in packages of 64K or less with each package preceded by a GPIB Data Pass Through command.

Byte #	Description
0	246 (x'F6') - GPIB data pass through
1	GPIBADD GPIB Address - This byte specifies the GPIB address over which data is sent/received. 0 <= GPIBADD < 31. QCR's default addresses are 2 and 3.
2	READ - Read/write flag - specifies the direction of the data transfer. = x'00' data is to be written to the GPIB device once. = x'01' data is to be written to the GPIB device three times. = x'FF' data is being read from the GPIB device.
3	INTERRUPT - Interrupt flag - specifies whether an SRQ interrupt is generated by the device receiving this data. = x'00' no interrupt anticipated. = x'FF' wait for interrupt before proceeding.
4 - 5	BYTECNT - Byte count specifying the number of bytes to be read or written. Treated as an unsigned positive integer. A zero value is interpreted as a request for 65536 bytes.
6 to BYTECNT+5	List of data byte(s) to be written to the GPIB device. When reading from the GPIB, this list is excluded. However, the SCODL processor waits to receive BYTECNT bytes from the GPIB and transmits these to the host.

Table 1 GPIB Data Pass Through Command Format

(247) Define Kerning Points

This command allows you to define the points at the edges of a macro character from which inter character spacing is measured. Kerning points set by this command are assigned to macros subsequently defined using opcode 227 (Define a Macro), until another opcode 247 is used or a reset occurs. If a macro is defined with kerning points of 0, or without kerning points having previously been defined, the macro is treated as a block the size of the virtual screen.

These kerning points allow externally defined character macros (see opcode 227) to be kerned (spaced) automatically according to a system that produces typographically attractive and legible character strings. Their use is introduced in the discussion of opcode 233.

Kerning points are defined as horizontal measurements from the center line of the virtual screen on which the macro is defined to points that represent bounds that no neighboring macro should extend. In addition to three kerning points on either side of the macro, cap line and base line measurements are used to state the distances from the center of the virtual screen to the cap line and base line for use if vertical kerning is required.

You must set left-hand side (LHS) and right-hand side (RHS) kerning points at three vertical levels on the characters, which are consistent from character to character, and must be specified in order. For example, LHS top, middle, and bottom; RHS top, middle, and bottom. Their vertical location is of no consequence to the MVP+, which simply finds the pair of kerning points - top, middle, or bottom - closest between a given character pair, and sets them apart by the specified inter character spacing. Refer to Figure 1, where the inter character spacing has been placed between the top pair of kerning points.

Byte #	Description	
0	247 (x'F7') - Define Kerning Points opcode	
1-2	Cap line measurement	
3-4	Base line measurement	
5-6	LHS 1	(These values are
7-8	LHS 2	in standard SCODL
9-10	LHS 3	coordinate format)
11-12	RHS 1	
13-14	RHS 2	
15-16	RHS 3	

Table 1 Define Kerning Points Command Format

Kerning point measurements are always positive values or zero. If the cap line, base line measurement, or all three kerning points on one side are set to zero, the kerning points are assumed to lie at the edges of the virtual screen. If only one or two of the kerning points on one side of a character are set to zero, they are ignored and the character is kerned from the points for which values are given.

> ***NOTE:*** *For accents such as those required for French text, zero spacing is required so that the accent appears in the same horizontal space as the character which it modifies. This can be accomplished by using x'FFFF' as the value for LHS 1.*

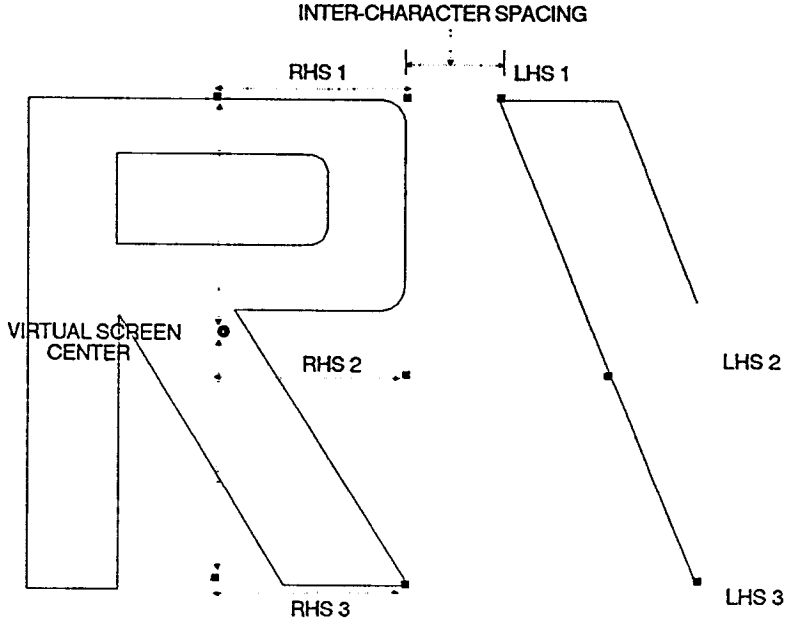


Figure 1 Kerning Points and Cap Line/Base Line Measurements

(248) Load Three Look-up Tables

This command loads the MVP+ with data for all three LUTs. Neutral LUTs are loaded using Opcode 251.

Byte #	Description
0	248 (x'F8') - Load three look-up tables
1	0 - zero byte (for future expansion)
2	RED0 - Contents of location 0 of Red LUT
3	RED1 - Contents of location 1 of Red LUT
.	. . .
.	. . .
257	RED255 - Contents of location 255 of Red LUT
258	GREEN0 - Contents of location 0 of Green LUT
259	GREEN1 - Contents of location 1 of Green LUT
.	. . .
.	. . .
513	GREEN255 - Contents of location 255 of Green LUT
514	BLUE0 - Contents of location 0 of Blue LUT
515	BLUE1 - Contents of location 1 of Blue LUT
.	. . .
.	. . .
769	BLUE255 - Contents of location 255 of Blue LUT

Table 1 Load Three Look-Up Tables Command Format

(249) Virtual Screen Maximum Dimensions

Subcommand 0

As explained in *Selecting Output Image* in Section 2, this command specifies the maximum dimension of the virtual screen in output pixels. Table 1 below illustrates the format for this command. The dimension is specified as a positive 16-bit integer between 2 and 32768. This subcommand is used if symmetric scaling is to be done. For example when XDIM=YDIM, there is no change in the proportion between height and width.

The values set by this command remain in effect until superseded by a subsequent code 249 command. This command must only be sent at the beginning of an image before the first geometric object descriptor. Object descriptions that precede this command do not appear in the final image.

On power up, the default virtual screen maximum dimension is x'0800' in 2K mode and x'1000' in 4K mode, with XDIM = YDIM.

Byte #	Description
0	249 (x'F9') - Virtual screen maximum dimensions
1	0 - Specify x/y dimension
2 - 3	DIM - Virtual screen maximum dimension 2 <= DIM < 32768 (DIM = XDIM = YDIM)

Table 1 Virtual Screen Maximum Dimension Command Format (x=y)

Subcommand 1

This subcommand is identical to the preceding one except that different X and Y dimensions may be specified. It is used if non-symmetric scaling is to be used; for example, when XDIMYDIM, thereby creating a change in the proportion between height and width is created.

Byte #	Description
0	249 (x'F9') - Virtual screen maximum dimensions
1	1 - Specify X and Y dimensions
2 - 3	XDIM - Virtual screen maximum X dimension
4 - 5	YDIM - Virtual screen maximum Y dimension

Table 2 Virtual Screen Maximum Dimensions Command Format (xy)

(250) Output Image Dimensions

This command specifies the portion of the virtual screen that is to be output in terms of output pixels. Section 5, *Output Image Selection* explains the relationship of output pixels to the virtual screen.

The output image is bounded on the left and right by first and second X coordinates, and on the top and bottom by first and second Y coordinates. All data within the virtual screen image area corresponding to these dimensions, including the bounds themselves, are output. The total number of pixels output per line are as follows:

X2 - X1 + 1.

The total number of scan lines output are as follows:

Y1 - Y2 + 1.

The values set by this command remain in effect until superseded by a subsequent code 250 command.

This command must only be sent at the beginning of an image before the first geometric object descriptor. Object descriptions, which precede this command, do not appear in the final image.

Byte #	Description	Defaults on Power-up	
		2K	4K
0	250 (x'FA') - Output image dimensions		
1	0 - zero		
2 - 3	X1 coordinate	x'FC00'	x'F800'
4 - 5	X2 coordinate	x'03FF'	x'07FF'
6 - 7	Y1 coordinate	x'02FF'	x'05FF'
8 - 9	Y2 coordinate	x'FD00'	x'FA00'

Table 1 Output Image Dimensions Command Format

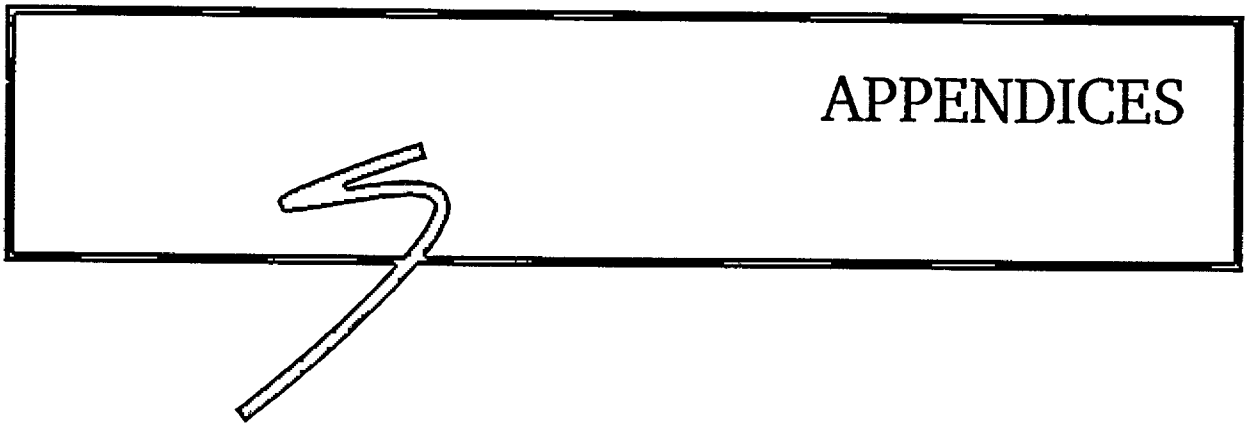
(251-254) Load LUTS Individually

Table 1 below illustrates the format for the Load Look-up Tables commands. Each command contains the appropriate opcode - Neutral, Red, Green, or Blue - and 256 LUT intensity values. The LUT is loaded in ascending order starting with LUT0 loaded into location zero, and proceeding through to LUT255 loaded into location 255.

Byte #	Description
0	Opcode = 251 (x'FB') Load the Neutral LUT = 252 (x'FC') Load the Red LUT = 253 (x'FD') Load the Green LUT = 254 (x'FE') Load the Blue LUT
1	0 - zero byte (for future expansion)
2	LUT0 - Contents of location 0 of LUT
3	LUT1 - Contents of location 1 of LUT
.	. .
.	. .
257	LUT255 - Contents of location 255 of LUT

Table 1 Load the LUTs Command Format

APPENDICES



This section contains:

- Appendix A: The SCODL Status Field
- Appendix B: Output Devices
- Appendix C: Errors and Messages
- Appendix D: Alphabetical List of Opcodes by Opcode Description

Appendix A: More on the SCODL Status Field

The SCODL Status field is an area of the MVP+ Display that appears while operating the Foreground program. The byte descriptions of this field are listed in Table A-1.

Status Report Format

The format of the SCODL Status Field is illustrated in Figure A-1 below:

SCODL STATUS

```
FNFNAAAABBBBCCCCDDDDEEEEFFFFGGGGHHHHIIIIJJJJKKKK
LLLLLLLLMMMMMMMMNNNNNNNNOOOOOOOOPPPPFNFNQQQQQQQQ
RRRRSSSSTTTTUUUUUUUUVVVVWWWWXXXXYYYYYYYYZZZZA1AZB1B1
C1C1D1D1E1E1F1F1G1G1H1H1
```

Figure A-1 Contents of the SCODL Status Field

Field Descriptions

The format of the SCODL Status Report message is shown in Table A-1. The message is returned by the MVP+ if either the SCODL status return is enabled during configuration, or if SCODL opcode 129 is received.

Bytes	Field	Description	
0-5	'! 00'	Status Message Header	
6-9	FNFN	Frame reference number of current image	
SCODL Table Sizes (Opcode 239) Selected:			
			For portions actually used, see bytes:
10-13	AAAA	NUMOBJ	86-89
14-17	BBBB	NWRKSLT	106-09
18-21	CCCC	NCHKSLT	110-13
22-25	DDDD	NPRIO	114-17
26-29	EEEE	NRLC	118-21
SCODL Virtual Screen Maximum Dimensions (Opcode 249):			
30-33	FFFF	XDIM	
34-37	GGGG	YDIM	
SCODL Output Image Locaters (Opcode 250):			
38-41	HHHH	X1	
42-45	IIII	X2	
46-49	JJJJ	Y1	
50-53	KKKK	Y2	

Table A-1 Status Report Field Descriptions (continued...)

Bytes (Continued...)	Field	Description
SCODL Meta-file Processing Data:		
54-61	LLLLLLLL	Size (bytes) of current image's SCODL meta-file processed thus far.
62-69	MMMMMMMM	Size (bytes) of edge record buffer for current image.
70-77	NNNNNNNN	Number of SCODL commands generated.
78-85	OOOOOOOO	Number of edge records generated.
86-89	PPPP	Number of object slots actually used.
Last Scan-converted Image Data:		
90-93	FNFN	Frame number of last scan-converted image.
94-104	QQQQQQQQ	Size (bytes) of RLC buffer generated for image.
102-105	RRRR	Number of scan lines processed.
106-109	SSSS	Number of work slots (Opcode 239) actually used.
110-113	TTTT	Number of check slots (Opcode 239) actually used.
114-117	UUUU	Number of priority elements (Opcode 239) actually used (2 * NPRIO elements were allocated).
118-121	VVVV	Number of RLC slots (Opcode 239) actually used.
122-125	WWWW	X-coordinate of first pixel drawn on output device (Opcode 236).
126-129	XXXX	Y-coordinate of first pixel drawn on output device (Opcode 236).
130-137	YYYYYYYY	Size in bytes of the storage area for user macros (Opcode 227).
138-141	ZZZZ	Number of user-defined macros (Opcode 227) in user macro storage area.
142-143	A1	Current output device number (Opcode 223).
144-145	A2	Current output device mode (Opcode 223).
146-149	B1B1	Current RLC box transparent color (Opcode 221).
Locations for SCODL RLC Merging Box (Opcode 220):		
150-153	C1C1	RX1
154-157	D1D1	RX2
158-161	E1E1	RY1
162-165	F1F1	RY2
166-169	G1G1	SCODL image background color code (Opcode 218)
170-171	H1H1	SCODL processing flags Of these two bytes, only the first six bits (15-10) are currently in use. Bits turned on (set to 1) indicate that for this image: 15 - Virtual memory was used to store RLC's 14 - GSF was enabled 13 - RLC storage mode was used 12 - LUT compensation was performed 11 - Thicken edge was enabled 10 - At least one object was actually thickened in the output image

Table A-1 Status Report Field Descriptions

Appendix B: Output Devices

The subsequent pages provide detailed information regarding the output devices and graphics adapters that are supported by MVP+ and SCODL. They are described in alphabetical order.

CAUTION: *Never connect or disconnect the Imapro Super Cable between the printer and the MVP+ while either device is powered up.*

Calcomp Plotmaster

The Plotmaster printer is supported by the MVP+ as device number 25 and is driven by the MVP+ using an Imapro Super Cable. The maximum image size that the MVP+ will print on the Plotmaster is 1600 x 2000 pixel lines. The driver only supports 3-ink operation.

Operation

Printing images on the Plotmaster involves the following steps:

1. Make sure the printer is turned on and connected to the MVP+ with a Super Cable.

2. Send the switch file SWICAL.SCD found in the \QVP\CONTROL directory. SWICAL.SCD specifies the following:

   DF 19 00 - select the Plotmaster as the current output device.
   EC 01 0064 0000 - the image will start 100 pixels from the left.
   EC 02 08 - use matrix size 8 for dithering.
   F9 00 073A - set the virtual screen maximum dimension to 1850.
   FA 00 FD44 02BB 039C FC63 - set the output image dimensions to center image (x1 = -700, x2 = 699, y1 = 924, y2 = -925)
   E2 03 - set virtual screen image rotation to 270 degrees.
   D3 02 - LUT compensation off.

3. Send the RGB look-up table to the MVP+ to specify the image colors. For example, \QVP\LUT\STDLUT.SCD

4. Send the SCODL image data to the MVP+. For example, \QVP\IMAGES\CLOWN.SCD. The MVP+ will begin to print the image on the Calcomp Plotmaster.

File Support

The following files are included in the MVP+ software to support the Calcomp Plotmaster printer.

\QVP\QVP.EXE	Foreground program
\QVP\QVP0_OVL.COM	Driver program
\QVP\QVP.DAT	SCODL process
\QVP\CONTROL\SWICAL.SCD	Switch file to select printer

Canon LBP-8 A2 Laser Printer

The Canon printer is supported by the MVP+ as device number 18 (decimal) and is driven by the MVP+ using an Imapro Super Cable. The maximum image size that the MVP+ prints on the Canon is 2337 pixels wide x 3324 lines long.

Operation

Printing images on the Canon involves the following steps:

1. Make sure that the printer is turned on and connected to the MVP+ with a Super Cable. The printer DIP switches should be in the factory set position except that ISO mode and full paint should be selected.

2. Send \QVP\CONTROL\SWICAN.SCD to the MVP+. This sets up the MVP+ for Canon output and needs to be done only once unless you want to change parameters. SWICAN.SCD specifies the following hexadecimal codes:

   DF 12 00 - select Canon as output device.

   EC 01 00 00 00 00 - the image is positioned at top left corner of page.

   EC 02 08 00 - use matrix size 8 for dithering.

   F9 00 0C FC - set virtual screen maximum dimension to 3324.

   FA 00 FB 70 04 90 06 7D F9 82 - set output image dimensions to x1 = -1168, x2 = 1169, y1 = 1661, y2 = -1662.

   E2 03 - virtual screen image rotation by 270 degrees.

   D3 02 - LUT compensation off.

3. Send a RGB look-up table such as \QVP\LUT\STDLUT.SCD to the MVP+ to specify color. As noted later the RGB data is reduced to shades of grey for printing on the Canon. This step needs to be done only once unless you want to change the palette that defines the association of SCODL color codes with RGB values.

4. Send SCODL image data to the MVP+. The MVP+ prints the images on the Canon.

Dithering and Shades of Grey

The MVP+ converts SCODL colors to shades of grey by the equation:

$$Y = .30R + .59G + .11B$$

where Y = luminance, and R,G,B are the red, green, blue intensities. These shades of grey are implemented on the Canon printer using the ordered dithering technique. With an 8 x 8 matrix, 65 levels of grey are supported.

Pass Through Command

This command is used to pass data directly between the host processor and the device attached to the Super Cable. It is identical to the SCODL GPIB pass through command (246 = F6h) except that bytes 1,2, and 3 of the command must be zero; only writes are allowed. Device 18 = 12h, specifying the Canon LBP must have been selected for this command to work.

This command may used to send commands to the LBP such as selecting the number of copies to print.

File Support

The following files are included with the MVP+ software to support the Canon LBP-8A2 printer.

\QVP\CONTROL\SWICAN.SCD	this file selects the Canon LBP output.
\QVP\CONTROL\BG255W.SCD	this file sets the background color to 255 and defines it as white. This is useful for printing images that would otherwise have a black background, but will not work if the background is covered by a SCODL polygon.
\QVP\DEMO\CANDEMO.BAT	this MVP+ Foreground batch file prints a number of Imapro test images on the Canon LBP-8 A2 printer.

Diablo C150 Ink-Jet Printer

The Diablo C150 printer is supported by the MVP+ as device number 3 (decimal) and is driven by the MVP+ by an Imapro Super Cable.

File Support

With the MVP+ software, Imapro supplies three switch files that contain output image dimensions for the Diablo printer. The three files correspond to three different image sizes:

\QVP\CONTROL\SWIDIABS.SCD	for a small output image.
\QVP\CONTROL\SWIDIABM.SCD	for a medium-sized output image.
\QVP\CONTROL\SWIDIABL.SCD	for a large output image.

If you want to modify the parameters provided by these files, refer to the descriptions of opcodes 236, 249, 250.

When the printer goes off-line for any reason, such as running out of ink or paper, the output device code displayed in box 17 of the MVP+ Status Field displayed in the Foreground operation flashes repeatedly. The MVP+ continues to process the image when the printer is put back on line.

The largest image that may be processed is one 1024 pixels wide and 32,768 lines long. This printer prints four rows of dots at once. If the number of lines in an image is not a multiple of four, some blank lines appear between it and any subsequent image.

Hewlett Packard LaserJet Series II

The Hewlett Packard LaserJet printer is supported by the MVP+ as device number 6 and is driven by the MVP+ using an Imapro Super Cable. The maximum image size that the MVP+ prints on the HP is 2550 pixels by 4200 lines long.

Operation

Printing images on the HP Laser involves the following steps:

1. Make sure that the printer is turned on and connected to the MVP+ with a Super Cable.

2. Send the switch file that corresponds to the size of paper being used to the MVP+. For example, \QVP\CONTROL\SWIHP11.SCD specifies the following settings:

DF 06 02 - select the HP Laser as the current output device using letter size paper

EC 01 00 3C 00 00 - start the image at position (60,0)

EC 02 08 00 - set the dither matrix size to 8

F9 00 0B B8 - set the virtual screen maximum dimension to 3000

FA 00 FB 82 04 7E 05 DC FA 24 - set the output dimensions to x1 = -1150, x2 = 1150, y1 = 1500, y2 = -1500

E2 02 - rotate image 90 degrees

D3 02 - no LUT comensation

3. Send the RGB look-up table that matches the image to the MVP+. For example, \QVP\LUT\STDLUT.SCD. This step is omitted if the RGB table is contained in the image SCODL file.

4. Send the SCODL image file to the MVP+. The MVP+ sends the image to be printed on the HP Laser printer.

Dithering and Shades of Grey

The method of dithering and shades of grey are similiar to the Canon LBP-8 A2 described earlier. For information, refer to the Canon description.

File Support

The following files are included in the MVP+ software to support the HP LaserJet:

\QVP\CONTROL\SWIHP11.SCD - for 8.5 x 11" paper

\QVP\CONTROL\SWIHP14.SCD - for 8.5 x 14" paper

\QVP\CONTROL\SWIHPA4.SCD - for A4 size paper

\QVP\CONTROL\SWIHPEX.SCD - for Executive (7.25 x 10.5") size paper

Each switch file contains to necessary SCODL commands to select the printer device, the start position, virtual screen and the output dimensions to center the image for landscape output. No compensation is used.

IBM Color Jet Printer 3852-2

The IBM Color Jet printer is supported by the MVP+ as device number 13 (decimal) and is driven by the MVP+ using an Imapro Super Cable.

Resolution is given as being 100 dpi (horizontal) x 96 dpi (vertical), but MVP+ treats pixels as being square. Therefore, images may appear to be slightly stretched vertically. Dithering is used to achieve multiple colors similar to other ink jets.

Printer Options

By using the <Esc> sequence, the MVP+ puts the printer into text mode at the beginning of each image. Therefore, subsequent printing of text may be modified.

The printer has a bold (double striking) mode that can be locked on by a printer switch or turned on and off by commands from the host computer. The MVP+ software does not issue any of these commands and works with the bold condition as is.

By not using the bold option, the printer saves ink and increases speed of output by about 60%. However, the image color is noticeably less dense.

Note: IBM recommends bold for transparencies.

File Support

The following files are included with the MVP+ software to support the IBM Color Jet Printer 3852-2.

SWIICJ.SCD	selects the IBM Color Jet printer at a resolution of 800x600 pixels.
SWIICJS.SCD	selects the IBM Color Jet printer at a resolution of 400x300 pixels.

IBM Enhanced Graphics Adapter

The IBM Enhanced Graphics Adapter (EGA) driving the Enhanced Color Display (ECD) is supported by the MVP+ as device number 10 (decimal). The IBM Graphics Memory Expansion Card must be installed in the system unit.

Resolution

Two horizontal resolutions are supported: 560 and 640 pixels per line. Horizontal resolution is selected through a configuration option or by using SCODL opcode 223 (Select output device) to set bit 0 of the output device mode byte to 0 for 560 pixels, or to 1 for 640 pixels.

Vertical resolution is the number of scan lines allowed by the SCODL output *viewport* (SCODL opcode 250); this is between 350 and 640 lines. For high resolutions, note these considerations:

- Vertical resolutions of more than 409 lines require additional memory on the graphics memory expansion card.
- For resolutions in the mid-400s and above, the display's vertical synchronization may fail.
- As vertical resolution increases, the display refresh rate decreases from 60 Hz at a resolution of 350 vertical lines. Flicker may also become apparent.

Configurations

The MVP+ software assumes that the EGA/ECD are already in 80 character (640 pixel) per line mode, except for the first EGA/ECD image output after MVP+ start up. This is intended to prevent an annoying flash of the display at the start of an image. If some other software leaves the EGA/ECD in 40 character (320 pixel) per line mode between images, problems are encountered by the MVP+ driver version.

The MVP+ achieves a wide range of colors on the EGA/ECD by *dithering*, a process that mixes pixels of different colors in dot matrix patterns.

Operation

Outputting images to the EGA involves the following steps:

1. Send file \QVP\CONTROL\SWIEGA.SCD or equivalent data to the MVP+. This sets up the MVP+ for EGA output and needs to be done only once unless you want to change parameters. SWIEGA.SCD specifies the following hexadecimal codes:

DF0A01 - select the EGA as output device

EC0100000000 - the image is positioned at the top left corner of the screen

EC020400 - use matrix size 4 for dithering

F9010280 OLD2 - set the virtual screen maximum dimension to 640 x 350

FA00FEC0013F00AEFF51 - set output image dimensions to X1= -320, X2= 319, Y1= 174, Y2= -175

E200 - virtual screen image rotation set to zero

D302 - LUT compensation off

2. Send a RGB look-up table to the MVP+ to specify color, for example, STDLUT.SCD.

3. Send SCODL image data to the MVP+. The MVP+ displays the image on the EGA screen.

File Support

The following file included in the MVP+ software supports the IBM Enhanced Graphics Adapter.

QVP\CONTROL\SWIEGA.SCD this file selects the EGA for output.

Lasercomp Printer

The Lasercomp printer is supported by the MVP+ as device number 19 (decimal) and responds to GPIB address 4 for both commands and data.

The IEEE interface provided by Monotype currently supports only a small subset of the full Lasercomp language. The actual command codes are completely transparent to the software generating images for the MVP+ to send to the printer, but are listed below to allow possible use via the standard GPIB pass-through command.

Code	Function
x'84'	End of job (expose)
x'AA'	Load picture file
x'D0'	Call picture
x'D8'	Set primary x coordinate
x'D9'	Set primary y coordinate

The picture data transmitted consists entirely of Monotype RLCs; no headers are required.

The data is encoded into 16-bit words with the most significant byte sent first. Each word can be one of two possible run-length formats:

Short	Long
bit 15 = 0	bit 15 = 1
bits 7 to 14 = white run length (0 to 255)	bit 14 = 0 for white, 1 for black
bits 0 to 6 = black run length (0 to 127)	bits 0 to 13 = run length

Each scan line must be terminated with a word of x'8000'. End of image is indicated by two successive words of x'0000'. An SRQ is reported by the Lasercomp on each of the following conditions:

After completion of a powerup sequence (the printer should be powered up before the MVP+ is started).

- After storing an image.
- After completing an exposure.
- After any error has been detected.

Lasercomp SRQ Codes

The following is a list of Lasercomp SRQ codes:

x'00'	Successful completion of operation
x'81'	GPIB handshake error

x'82'	Illegal command
x'83'	Error opening picture file
x'84'	Disk full
x'85'	Error closing picture file
x'86'	Error writing to picture file
x'87'	Picture file too large
x'88'	Too many pictures selected for current exposure
x'89'	Error creating job file

Current SRQ error codes might be ambiguous with QCR/PCR errors - the I/O processor status code for a GPIB-SRQ has the same primary code for both Lasercomp and QCR/PCR. The secondary code has a different meaning depending on the device. The Lasercomp printer **must** be reset after any errors are reported.

Operation

Exposure of Lasercomp images involves the following steps:

1. [DF 13 00] - select Lasercomp as output device.

2. [EE ff ff] - set frame number. This frame number is used as the Lasercomp picture number for storing the image on the Lasercomp disk (a [AA ff ff] command preceeds the image data).

3. [.....] - transmit SCODL image to MVP+. The MVP+ sends Lasercomp RLCs to the printer.

4. Repeat steps 2 and 3 for each image that will be put on the next exposure. Up to 16 images can be combined in this manner.

5. [F6 aa 03 00 bbbb xxxx yyyy pppp xxxx yyyy pppp ....] - expose group. This is a device pass through command used to expose one or more images as a single layout where:

aa = GPIB device address
bbbb = byte count for picture information
xxxx = primary horizontal address for picture
yyyy = primary vertical address for picture
pppp = picture number

There is a (xxxx yyyy pppp) triplet for each picture to be exposed in this layout. The actual data sent to the Lasercomp is:

```
[D8 xxxx D9 yyyy D0 pppp]    ; call up first picture
[D8 xxxx D9 yyyy D0 pppp]    ; call up second picture
                .
                .
                .
[84]                ; end-of-job (start exposure)
```

When images are being transmitted to the Lasercomp, the spooler IN/OUT numbers flashes the current frame number (picture number). This frame number is returned in an end-of-image message for that image. When an exposure is started, the exposure number flashes in the spooler OUT position. After completion, the exposure number is returned in an end-of-image message. When the MVP+ is started up, the exposure number is set to x'8000' and increments with each exposure.

Limitations

When using the Lasercomp printer, there are several limitations that should be noted. Read the following carefully before attempting an output.

- If the standard VMEM is used to handle RLC editing, the output image is limited to 1 megabyte in size. If the special VMEMLSR is used, the limit is 20 megabytes.
- A maximum of 16 images can be combined on one exposure.
- The MVP+ must be operating in RLC-editing mode.
- The MVP+ frame number is used as the picture number for storing images on the Lasercomp disk. Therefore, the controlling software must manipulate the frame number to make sure the disk does not get filled. If an image will not be used again after it has been exposed, the same frame number should be used for the next exposure.
- The Lasercomp is currently supported only in black/white mode - dithering will not be done to produce grey-shades. The current background color (color 0 by default) is used to represent white. Any other color code represents black.

Matrix Thermal Transfer 200 Copier

The Matrix TT200 printer is supported by the MVP+ as device number 12 (decimal) and is driven by the MVP+ using an Imapro Super Cable. The largest image that can be output on the TT200 is 1568 pixels wide and 1920 lines long.

Operation

The Matrix TT200 thermal copier's resolution is 200 dots per inch. With the MVP+ software, Imapro supplies two files that contain output image dimensions for this printer. The two files correspond to two different output image sizes:

\QVP\CONTROL\SWITTM.SCD	for a medium-sized output image.
\QVP\CONTROL\SWITTL.SCD	for a large output image.

If you want to modify the parameters provided by these files, refer to the descriptions of opcodes 236, 249, 250.

IMPORTANT: *The TT200 printer provides two possible printing modes: fine or standard. Standard mode can be used if your image contains only the eight basic colors or if you need a quick printout. Fine mode prints up to 256 colors and provides the best possible color quality but takes up to 140 seconds compared to up to 60 seconds in standard mode.*

It is recommended that you use fine mode, especially when dithering. Lift up the Operation Cover on the front right-hand side of the unit and check to make sure that the selector switch selects **FINE.**

Dot Matrices and Color Look-Up Tables

Ink-jet and thermal copiers typically have 3 or 4 colors of ink - cyan, magenta, yellow, and possibly black. Combinations of these inks are used to create the 8 basic colors - red, green, blue, cyan, magenta, yellow, white, and black. All colors can be specified as mixes of red, green, and blue in the look-up tables (see opcode 248) and colors that are not one of the eight basic colors are printed as patterns of dots of those eight colors. The square arrangement of dots that is repeated throughout the shade is called the dot matrix. Outline colors should be chosen from the eight basic colors to avoid a speckled appearance along thin lines.

A subcommand of opcode 236 is used to specify the size of the dot matrix. Include this command in your image file if you do not use Imapro supplied switch files. See the description of this opcode in Section 8, *SCODL Opcodes* for more information.

Each LUT value is converted to a dot matrix, according to the following formula. Given a LUT value between 0 and 255, for example red, called *r*:

$r/255 \times n^2$ (rounded off)

This formula indicates the number of dots in a matrix of n2 dots, where n = 1, 2, 4, or 8, which will reflect, transmit, or radiate red (i.e. white, red, yellow, or magenta).

Output colors show the following characteristics:

1. Color is generally linear with respect to LUT values, but this linearity is affected to some extent by an overlap of dots.

2. The number of different shades of any one component (red, blue, or green) is n2 + 1, where n2 is the total number of dots in the matrix. The total number of possible colors is the cube of this value. For example,

   In a 1 x 1 matrix, n = 1: (12 + 1)3 = 8
   In an 8 x 8 matrix, n = 8: (82 + 1)3 = 274,625

3. Due to limitations inherent in the output, the fraction of dots in the matrix that reflect a given component may not equal the fraction of that component that is actually reflected. The user of a thermal copier may disable all shades other than the eight basic colors, either by specifying a dot matrix size of 1, or by loading look-up tables which contain only **all** or **nothing** color components; all LUT values being either 0 or 255.

Mitsubishi G650 Thermal Printer

The Mitsubishi G650 printer is supported by the MVP+ as device number 7 and is driven by the MVP+ using an Imapro Super Cable. The maximum image size the MVP+ prints on the G650 is 3392 pixels wide x 4473 lines long. Since this size is bigger than any standard format, the supplied switch files specify smaller dimensions.

The switch files support 3 ink (cyan, magenta, yellow) or 4 ink (cmy + black) operation. In 3 ink operation, black dots are generated by overprinting cyan, magenta, and yellow inks. In 4 ink operation, black dots are generated by overprinting cyan, magenta, yellow, and black inks.

Operation

Printing images on the G650 involves the following steps:

1. Make sure the printer is turned on and connected to the MVP+ with a Super Cable. We recommend that you use light intensity. You must power down the G650 before you reselect another intensity.

2. Send a switch file to the MVP+ e.g. \QVP\CONTROL\SWIG65A3.SCD This sets up the MVP+ for G650 output and needs to be done only once unless you want to change parameters. SWIG65A3.SCD specifies the following:

DF 07 00 - select G650 as output device, mode = 0. This is the 3 ink mode. Mode = 1 selects 4 ink output.

EC 01 00 00 00 00 - the image is positioned at top left corner of page.

EC 02 08 - use matrix size 8 for dithering

F9 00 10 EE - set virual screen maximum dimension to 4334.

FA 00 F9 60 06 9F 08 76 F7 89 - set output image dimensions to x1 = -1696, x2 = 1695, y1 = 2166, y2 = -2167.

E2 03 - virual screen image rotation by 270 degrees.

D3 02 - LUT compensation off.

3. You may want to send a color compensation file to the MVP+ e.g. G650-3.SCD This file adjusts the color balance of the printed output and needs to be done only once. G650-3.SCD specifies the following:

D3 01 - turn on compensation.

D7 02 01 ...768 bytes... - compensate using these 768 bytes of look-up table.

4. Send a RGB look-up table to the MVP+ to specify colors e.g. \QVP\LUT\STDLUT.SCD

5. Send SCODL image data to the MVP+ e.g. \QVP\IMAGES\CLOWN.SCD The MVP+ prints the image on the Mitsubishi G650.

Files Support

The following files are included in the MVP+ software to support the Mitsubishi G650 Thermal Printer:

\QVP\CONTROL\SWIG65A3.SCD - selects G650 A3 media (3392 x 4334), 3 ink.
\QVP\CONTROL\SW4G65A3.SCD - selects G650 A3 media, 4 ink.
\QVP\CONTROL\SWIG65A4.SCD - selects G650 A4 media (2368 x 2882), 3 ink.
\QVP\CONTROL\SW4G65A4.SCD - selects G650 A4 media, 4 ink.
\QVP\CONTROL\SWIG65B.SCD - selects G650 B media (3200 x 4472), 3 ink.
\QVP\CONTROL\SW4G65B.SCD - selects G650 B media, 4 ink.
\QVP\CONTROL\SWIG65A.SCD - selects G650 A media (2432 x 2672), 3 ink.
\QVP\CONTROL\SW4G65A.SCD - selects G650 A media, 4 ink.
\QVP\CONTROL\SLOW-3.SCD - selects G650, 3 ink. Slows data transfer rate.
\QVP\CONTROL\SLOW-4.SCD - selects G650, 4 ink. Slows data transfer rate.
\QVP\LUT\G650-3.SCD - a LUT compensation for 3 ink mode.
\QVP\LUT\G650-4.SCD - a LUT compensation for 4 ink mode.

Number Nine Revolution Graphics Adapter

The Revolution adapter is supported by the MVP+ as device number 21 (decimal).

File Support

The MVP+ software contains two files that may be used with the Revolution adapter. These are as follows:

\QVP\DEMO\N9DEMO.BAT	demonstration batch file which processes several images and tests the board.
\QVP\CONTROL\SWIN9.SCD	switch file that sets up the MVP+ for the Revolution board.

The contents and descriptions of the switch file are as follows:

DF 15 00 - select Revolution board as output device, mode 0.
EC 01 0000 00 00 - output device dependent command. Specify start pixel.
F9 01 02 00 02AA - specify virtual screen maximum dimensions by indicating x/y dimensions for the virtual screen (512 x 682). This corrects for the 4/3 pixel aspect ratio of the Number Nine.
FA 00 FF 00 00 FF 00 EF FF 10 - specify x and y coordinates of the output image on the virtual screen. x1 = -256, x2 = 255, y1 = 239, y2 = -240 (512x480)
E2 00 - set virtual screen image rotation to 0 degrees.

D3 02 - turn off LUT compensation and use originally loaded LUT.

Targa 16, 24, 32 Graphics Adapter

The Targa 16, 24, 32 are supported by the MVP+ as device number 20 (decimal). The MVP+ operates only on the memory of the Targa. A program must initialize the Targa registers enabling the contents of Targa memory to appear on the display.

The AT&T program TRUEINIT or the Imapro supplied TARGAINI can be used to initialize the registers. For TARGAINI to operate properly the TARGA and TARGASET environment variables must be set. TARGA, TARGASET are discussed in AT&T documentation.

Operation

Displaying images using the Targa involves the following steps:

1. Make sure the display is turned on and connected to the output connector of the Targa. The board must be set up to display its memory. Use the TARGAINI command to set the environment variables by entering at the C> prompt:

```
C>CD IMAPRO
C>TARGAINI
```

2. Send \QVP\CONTROL\SWITG482.SCD or equivalent data to the MVP+. This sets up the MVP+ for Targa output and needs to be done only once unless you want to change parameters. SWITG482.SCD specifies the following:

DF 14 00 - select Targa 24 as output device, mode=0. Mode bit 0 can be set to 1 to turn on tansparency mode. In this mode, all data with SCODL color 255 is treated as transparent and does not modify the Targa display. Mode bit 1 can be set to 1, enabling the MVP+ to leave the Targa memory enabled. Normally it is disabled after the MVP+ puts up the image.

EC 01 00 00 01 E1 - the image is positioned at (0,481) in Targa display coordinates, which is the top left corner if a 482 line display is selected. For the Targa, (0,0) is the bottom left corner of the display. Note that this coordinate system for image placement is different from the centered system used for QCRs or the (0,0) = top left system used for most other MVP+ devices.

F9 01 02 00 02 24 - set virtual screen maximum dimensions: XDIM = 512, YDIM = 548. These numbers allow for the Targa's non-square pixel, which has width/height = 1.07143. This is without the overscan option; if one is used then YDIM should be: 512 x 1.07143 x 1.2 = 658 (decimal) = 292 (hex).

FA 00 FF 00 00 FF 00 F0 FF 0F - set output image dimensions to x1 = -256, x2 = 255, y1 = 240, y2 = -241 (512 x 482).

E2 00 - virtual screen image rotation by 0 degrees.

D3 02 - LUT compensation off.

3. Send a RGB look-up table to the MVP+ to specify color such as \QVP\LUT\STDLUT.SCD. This needs to be done only once unless you want to change the combination of SCODL color codes with RGB values called a palette.

Note: A compensation LUT might also be sent to adjust the colors displayed; Imapro does not currently supply one.

4. Send SCODL image data to the MVP+ and display the images via the Targa.

File Support

The following files are included in the MVP+ software to support the Targa 24

\QVP\CONTROL\SWITG482.SCD	this file selects the Targa at 512 x 482, and a pixel aspect ratio width/height = 1.07143.
\QVP\CONTROL\SWITG048.SCD	selects the Targa at 512 x 482, and a pixel aspect ratio width/height = 1.286. This is appropriate for a Targa using the overscan option.
\QVP\CONTROL\SWITG400.SCD	selects the Targa at 512 x 400, and a pixel aspect ratio width/height = 1.07143.
\QVP\CONTROL\SWITG040.SCD	selects for Targa output at 512x400, and a pixel aspect ratio width/height = 1.286. This is appropriate for a Targa using the overscan option.
\QVP\DEMO\TARGADEM.BAT	this MVP+ Foreground batch file displays a number of images via the Targa 24.

Tektronix 4692 Ink Jet Printer

The Tektronix 4692 color ink jet printer is supported by the MVP+ as device numbers 15 and 16 (decimal). Device 15 is used to produce paper copies and device 16 is used to produce transparencies.

Configuration

When selecting the default output device during MVP+ configuration, **K** selects the 4692 in paper output mode and **T92** appears in the device box in the Foreground mode. **Y** selects the 4692 in transparency mode and **92T** appears in the device box in the Foreground mode.

In Transparency mode, data is copied to the 4692 in two passes to avoid bleeding of the red, green, and blue pixels. Also, black, cyan, magenta, and yellow are double printed for improved density.

Operation

Printing images on the Tek 4692 involves the following steps:

1. Make sure the printer is turned on and connected to the MVP+ with an Imapro Super Cable.

2. Send file \QVP\CONTROL\SWIT92.SCD or equivalent to select the 4692 for paper output or file SWIT92T.SCD for transparencies. This setup of the MVP+ for Tek 4692 needs to be done only once unless you want to change parameters. SWIT92.SCD and SWIT92T.SCD specify the following:

DF0F00 - selects Tek 4692 in paper mode.

DF1000 - selects Tek 4692 in transparency mode.

EC0100000000 - the image is printed in the upper left corner of the page.

EC020800 - use a matrix of size 8 for dithering.

F9000600 - set virtual screen maximum dimension to 1536.

FA00FD0002FF023FFDC0 - set output image dimensions to x1=-768, x2=767, y1=575, y2=-576.

E200 - virtual screen image rotation set to zero.

D302 - LUT compensation off.

3. To compensate the colors of the Tek 4692, files \QVP\LUT\COMPT92.SCD and \QVP\LUT\COMPON.SCD may be sent.

4. Send a RGB look-up table, such as \QVP\LUT\STDLUT.SCD, to the MVP+ to specify color.

5. Send SCODL image data to the MVP+. The MVP+ prints the images on the Tek 4692.

File Support

The following files are included in the MVP+ software to support the Tektronix 4692:

\QVP\CONTROL\SWIT92.SCD	- Tek 4692 output in paper mode.
\QVP\CONTROL\SWIT92T.SCD	- Tek 4692 output in transparency mode.
\QVP\CONTROL\SWIT92R.SCD	- Tek 4692 rotated format output on paper.
\QVP\CONTROL\SWIT92RT.SCD	- Tek 4692 rotated format output on transparency.

Appendix C: Coded Message Descriptions

Messages returned by the MVP+ to the Message Field carry a variety of information. They indicate errors during MVP+ processing, output device configurations, and the successful completion of an image. Every message carries the frame number of the image to which it applies.

All coded messages are encoded in 7-bit ASCII and have the format shown in Figure C-1 below. If the prompt indicates **"Meta-file Ignored"**, the MVP+ does not process the image It goes on to process the next meta-file or input image provided one is loaded. As an alternative to looking up message meanings in this section, refer to the *MVP+ User's Guide* for a supplement on reading messages.

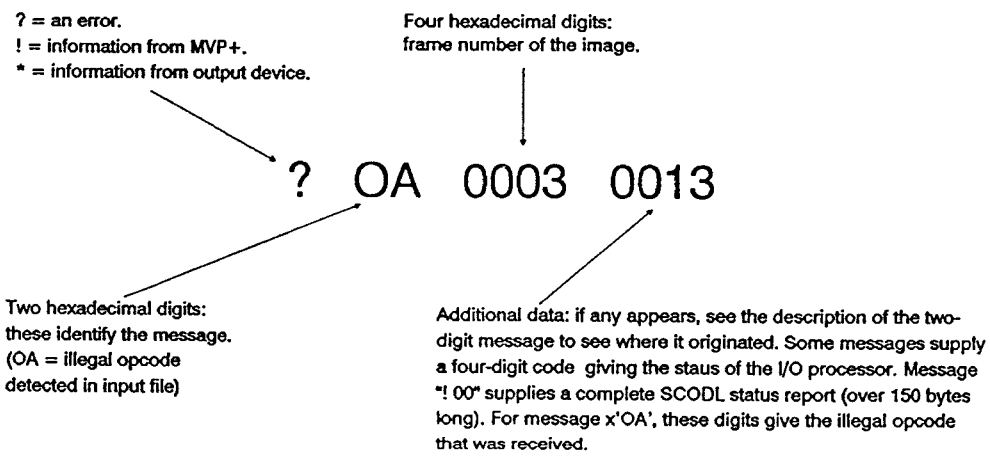


Figure C-1 Format of MVP+ Coded Messages

A list of all coded messages concerning MVP+ operations are listed in Table C-1.

Message	Description
! 00 FNFNX...X	SCODL Status Report Message. X...X is the SCODL status report.
? 01 FNFN	Input image overflow. NUMOBJ is too small. Use opcode 239 (load TABLBIG.SCD) to enlarge NUMOBJ. Meta-file ignored.
? 02 FNFN	Insufficient memory to allocate all edge records. More memory needed to process all of image's vectors (edges). Small overflows can be corrected by reducing memory allocations to buffers and tables; otherwise, system memory must be expanded to process this image. Meta-file ignored.
? 03 FNFN	RLC work table overflow. NWRKSLT too small. Use opcode 239 (load TABLBIG.SCD) to enlarge NWRKSLT. Meta-file ignored.
? 04 FNFN	Edge check table overflow. NCHKSLT too small. Use opcode 239 (load TABLBIG.SCD) to enlarge NCHKSLT. Meta-file ignored.
? 05 FNFN	Priority element buffer overflow. NPRIO too small. Use opcode 239 (load TABLBIG.SCD) to enlarge NPRIO. Meta-file ignored.
? 06 FNFN	Scan line RLC description overflow. NRLC too small. Use opcode 239 (load TABLBIG.SCD) to enlarge NRLC. Meta-file ignored.
? 07 FNFNXXXX	Wrong number of pixels (XXXX) calculated for scan line. An illegal viewport dimension may have been specified; a macro may extend beyond the virtual screen's edge; or thickening (opcode 216) has caused a macro to extend over the edge of the virtual screen. Meta-file ignored.
? 08 FNFN	Not enough memory to allow the SCODL table sizes requested. Table sizes (SCODL opcode 239) have all been reset to default values. Any user defined macros have been deleted. Meta-file ignored.
? 09 FNFN	Image RLC buffer overflow. Entire image RLC description could not be stored in memory. If error persists, memory must be enlarged to process the image. Meta-file ignored.

Table C-1 Coded Messages (continued...)

? 0A FNFNXXXX	Illegal SCODL opcode (XXXX) in input image file. Meta-file ignored. This error may be caused by an extra or missing byte in a SCODL command, putting data out of sync. Look at SCODL Status Field to find number of bytes processed, giving the location of the illegal opcode in the meta-file.
? 0B FNFNXXXX	Bad byte received during image input. (XXXX is an I/O processor error code*). Meta-file ignored. Check data format; ensure all characters sent are valid in the data format specified.
? 0C FNFNXXXX	Error on GPIB pass through command (XXXX = I/O processor error status*). Check cable connections.
*** 0D FNFNX...X**	Data read back from GPIB. 'X...X' is a data string of variable length, translated by the I/O processor into ASCII.
? 0E FNFNXXXX	Illegal interrupt (68000 vector #XXXX) received by SCODL processor; processing stopped. MVP+ is in an unknown state. Re start the MVP+. If error message persists, then MVP+ requires servicing.
? 0F FNFNXXXX	Illegal SCODL macro code (XXXX) in input file. Macro ignored. Look at SCODL Status Field to find number of bytes processed, giving the location of the illegal macro code in the meta-file. Check that FONTnn.BIN and FONTnn.SCD reference the same font "nn".
? 10 FNFNXX	Macro codes nested too deeply (XX levels of nesting exist). Macro request ignored. Arrange macro definitions with fewer macros-within-macros.
? 11 FNFN	End of macro request (opcode 241) when no macro active. Meta-file ignored. Any "end of macro" code (241) that does not follow a "define macro" code (227) must be deleted.
? 12 FNFNXXXX	Illegal function code sent to I/O processor by SCODL processor (XXXX = I/O processor error status*). SCODL processor is in an unknown state. Make a record of the error in case of a recurrence, and re-start the MVP+. If the error persists, then MVP+ requires servicing.

* See Table C-2 for I/O processor status codes.

Table C-1 Coded Messages (continued...)

? 13 FNFNXXXX	Bad transfer of image RLCs to disk (XXXX = I/O processor error status*). MVP+ may be out of disk space or may have encountered a disk fault.
? 14 FNFNXXXX	Bad end of image status returned from output device (XXXX = I/O processor error status*). Check output device and its connections to the MVP+.
! 14 FNFN0000	Image successfully completed on output device.
? 15 FNFNXXXX	Illegal output device number specified in an opcode 223 command (XXXX = output device number and mode, as used in opcode 223). Opcode 223 has been ignored.
? 16 FNFNXXXX	Input file could not be stored (XXXX = I/O processor error status*). Meta-file is too large to handle and was ignored. Meta-file size must be reduced or meta-file storage area expanded.
? 17 FNFNXXXX	Input file could not be retrieved from I/O memory (XXXX = I/O processor error status*). MVP+ is in an unknown state. Re start the MVP+. If error message persists, MVP+ requires servicing.
? 18 FNFNXXXX	Illegal spacing mode or text alignment (opcode 233) specified. XXXX = bytes 1 and 2 of opcode 233 command. Default mode loaded. Check usage of opcode 233 in meta-file.
? 19 FNFN	Macro string too wide for virtual screen. String ignored. Arrange macro string within virtual screen edges.
? 1A FNFNXXXX	Macro definition does not fit in memory. XXXX = macro code of failed definition. Macro definition must be made more concise, or some other macro(s) must be deleted. Check the SCODL Status Field for space currently allocated to macros. Macro definition ignored.
? 1B FNFNXXXX	Illegal SCODL opcode included in a macro definition. XXXX = illegal opcode. Meta-file ignored.

* See Table C-2 for I/O processor status codes.

Table C-1 Coded Messages (continued...)

? 1C FNFNXXXX	Nested macros. A "define macro" command has been included in a macro definition. XXXX = macro code of failed definition. Meta-file ignored.
? 1D FNFNXXXX	Invalid macro code or macro family code. Macro code must fall between 1 and 254; family code must fall between 1 and 127 inclusive. XXXX = macro code of the failed definition; macro definition ignored. Check usage of opcode 227.
? 1E FNFNXXXX	Macro definition exceeds 64 kilobytes. XXXX = macro code of failed definition; macro definition ignored. Definition must be made more concise.
? 1F FNFN	String too long to fit in temporary buffer area. String ignored.
? 20 FNFNXXXX	Error while reading an RLC image in virtual memory (XXXX is the I/O processor error status; see following table). The current image was aborted.
? 21 FNFNXXXX	Error while writing to an RLC image in virtual memory (XXXX is the I/O processor error status; see following table). The current image was aborted.
? 22 FNFNXXXX	Illegal RLC received (XXXX is the illegal 2-byte RLC) while inputting RLC merging box data (opcode 219). The rest of the RLC merging box data has been ignored, and replaced with transparent RLCs.
? 23 FNFN	RLC merging box data (opcode 219) command was received while MVP+ was not in RLC storage mode. The entire RLC merging box has been ignored.
? 24 FNFNXXXX	End-of-image-description marker (opcode 128) was detected while a macro was being defined. The macro definition (XXXX) has been ignored.
? 25 FNFNXXXX	A flood command (opcode 7) was received while the MVP+ was not in RLC storage mode (opcode 224, subcommand 1). The flood command has been ignored. XXXX is the flood color code.
? 26 FNFNXXXX	The seed point for a flood command (opcode 7) was not within the output image. The flood command has been ignored. XXXX is the flood color code.

Table C-1 Coded Messages (continued...)

? 27 FNFNXXXX	There may not have been enough memory to complete a flood command. This message is only a warning; processing has not been interfered with. Output the image and see if flooding has been fully performed. If not, you must enlarge the free memory in order to flood this image as specified. Do this by either reducing SCODL table sizes, reducing the number of defined macros, or by physically expanding MVP+ memory. XXXX is the flood color code.
? 28 FNFNXXXX	An illegal endpoint style (a subcommand of opcode 216, Thicken Edge) was selected. The thicken command was ignored. See opcode 216 in the *Programmer's Guide to SCODL* and use an allowed option. XXXX is (216) followed by the illegal subcommand used.
? 2A FNFNXXXX	The kerning table (opcode 214) carries a font code (XXXX) that is outside the allowable range of 1-127.
? 2B FNFNXXXX	An illegal macro file index (XXXX) has been specified in a Macro Access Mode command (opcode 206). The index must be in the range 0 through 255 inclusive. The 206 opcode is ignored and the SCODL processor is left in memory mode (0).
? 2C FNFNXXXX	There are no macro descriptions at the specified macro file index (XXXX) (opcode 206). The 206 opcode is ignored and the SCODL processor is left in memory mode (0).
? 2D FNFN	Stack underflow while processing a spline command (opcode 9, 10, or 11). Indicates a bug within the SCODL processor. The current image is aborted.
? 2E FNFN	Coordinate stack underflow while processing a spline command (opcode 9, 10, or 11). Indicates a bug within the SCODL processor. The current image is aborted.
? 2F FNFN	Illegal internal data type while processing a spline command (opcode 9, 10, or 11). Indicates a bug within the SCODL processor. The current image is aborted.

Table C-1 Coded Messages (continued...)

? 30 FNFNXXXX	Virtual Memory Macro read error during an opcode 206 command. The Font Identification codes of the Font Pointer Table (FPT) and the Macro Pointer Table (MPT) are different. Macros cannot be read from virtual memory, SCODL remains in macro access mode zero. XXXX indicates the font that was requested by the opcode 206 command.
? 31 FNFNXXXX	While in Virtual Memory macro access mode, a Macro definition (JIS code = XXXX) greater than 2048 bytes was requested. The image was too large to process and was ignored.
? 32 FNFNXXXX	Virtual Memory Macro read error during a macro or a string request. A Virtual Memory macro (JIS code = XXXX) identification code was found to be incorrect. The macro request was ignored.
? 33 FNFNXXXX	Illegal macro request while in Virtual Memory Macro access mode. There is no definition for the specified macro request (JIS code = XXXX), and the macro request has been ignored.
? 34 FNFNXXXX	Read error during Virtual memory read. XXXX is the returned I/O status. The image was aborted.
? 35 FNFNXXXX	Virtual Memory Macro request code (XXXX) out of range. The supplied macro code was not a legal word processor code, and therefore, an index into the macro descriptions could not be calculated. The macro request is ignored.
? 36 FNFNXXXX	Illegal virtual screen maximum dimension specified (opcode 249). XXXX is greater than 32767 (x'7FFF'). Virtual screen maximum dimensions ignored. Current image aborted. Reduce maximum dimension and try again.
? 37 FNFNXXXX	VMEM kerning table ID code mismatch. Indicates corruption of VMEM font data structure. XXXX is ID code read from VMEM.
? 38 FNFNXXXX	Kerning table exceeds maximum size of 6K. XXXX specifies the size of the kerning table. Requested font cannot be kerned—table is too large.
? 39 FNFN	Object file not completed. At least one object in the image is located off the edge of the virtual screen. Interior filling has not been turned off. Distortion may occur due to macro string or thickened edge extending beyond the virtual screen. Output of image still occurs. Use virtual screen clipping mode (opcode 224, subcommand 5) to remove distortion.

Table C-1 Coded messages

Input/Output Error Codes

The following codes listed in Table C-2 appear as the last two bytes of an error message. These codes indicate the status of the MVP+'s input/output processor.

Code	Description
0000	No error.
0001	QCR/PCR is not responding. It may not be set up for the correct device address. Check the film recorder's address programming and cable connections.
0004/005/006	QCR/PCR SRQ error. Reset film recorder; if error persists, re start the MVP+. If the error still persists, call a qualified service representative.
0007	IEEE handshake failure. QCR/PCR is not responding. Check the QCR DIP switches and cable connections.
XX15	Unsupported output device. XX = illegal device number.
XX20	Illegal request from SCODL processor. Re-start the MVP+. If error reoccurs, MVP+ disk files may be corrupted or the MVP+ may require servicing. XX = the illegal function.
XX21	Disk I/O error; occurred when IBM system's disk was accessed. XX is a DOS error code. These codes are: '0021' - Out of disk space; '0121' - Invalid function number; '0221' - File not found; '0321' - Path not found; '0421' - Too many open files (no handles left); '0521' - Access denied; '0621' - Invalid handle; '0721' - Memory control blocks destroyed; '0821' - Insufficient memory; '0921' - Invalid memory block address; '0A21' - Invalid environment; '0B21' - Invalid format; '0C21' - Invalid access code; '0D21' - Invalid data; '0F21' - Invalid drive was specified; '1021' - Attempted to remove the current directory; '1121' - Not same device; '1221' - No more files.

Table C-2 I/O Error Codes (continued...)

XX22	QCR/PCR error. XX is a QCR/PCR error code; see QCR/PCR documentation for detailed error information. '0022' - Usually indicates excess data received; '0122' - QCR/PCR is out of film; '8122' - Unexpected command byte; '8222' - Unexpected data byte; '8322' - Illegal command byte; '8422' - Illegal data byte; '8522' - Command interrupted; '8622' - Unimplemented function; '9122' - Memory fault; '9222' - GPIB interface fault; '9322' - CRT calibration fault - call for service; 'A122' - module fault - check module installation; 'A222' - filter wheel fault - reset the QCR.
XX23	Unrecoverable error occurred during output of an image to a thermal printer. The image has been aborted. Correct any error condition in the output device and check its cable connections. XX is a byte describing the status of the output line: bit 4 = 1 if the device is off line (e.g x'10'); bit 3 = 1 if paper is empty (e.g. x'08'); bit 2 = 1 if fault status line is active (e.g. x'04'). The user can ignore bits 1 and 0.
0024	A software or system error occurred while expanding an RLC image to a pixel image. Re-start the MVP+. If error reoccurs, disk files may be corrupted or the MVP+ may require servicing; software and/or hardware must be inspected by service personnel.
0025	SCODL output image dimensions (opcode 250) and start pixel coordinates (opcode 236) specify an image that is too wide or long for the output device.
0026	Error while trying to access virtual memory (VMEM). It is possible that the VMEM driver is not installed or the VMEM partition does not exist.
0030-003F	Error detected while receiving image data over the RS232 port. Refer to the IBM technical reference manual on this interface. 0011 abcd (for a, 1 = break; b, 1 = framing; c,1 = parity; d, 1 = overrun).
0081	Illegal memory access. Re start the MVP+; if message persists, MVP+ disk files are corrupted or the MVP+ may require servicing.

Table C-2 I/O Error Codes (continued...)

0082	Not enough memory to store entire input file; this code accompanies, and is the same as, a "? 16" MVP+ message.
XXB0	Illegal character (XX). If the input file contains control characters, the "ignore illegal characters" option should have been selected during configuration.
00B1	Input buffer overflow. Ensure that the same handshake mode has been selected for the MVP+ and the host computer.

Table C-2 I/O Error Codes

Appendix D: Alphabetical List of Opcodes by Opcode Description

Table D-1 provides a complete list of all opcodes in alphabetical order by their opcode name. This list is designed to give you a quick reference of the opcodes. Read Section 4, *Description of Opcodes* for more detailed information on each opcode.

Dec	Hex	Opcode	Bytes	Contents
234	EA	Begin a macro string	n	Macro requests, x'FF'
230	E6	Catenate macro string	n	Macro requests, x'FF'
8	08	Compressed mode polygon	n	One-byte differential coord. pts.
227	E3	Define a macro	3	Family/font, macro code, SCODL data
247	F7	Define kerning points	17	Cap line, baseline, 6 x coordt. values
237	ED	Delete macro family/font	3	Zero byte, family/font code
128	80	End of image description marker	2	Zero byte
241	F1	End of macro definition	1	---
3	03	Filled arc	18	Highlt, 0 byte, fill, width, 3 coord. pts.
7	07	Flood	6	Flood color, seed pt. X, Y coords.
246	F6	GPIB data pass through	n	Address, read, interrupt, bytecount, data
5	05	Hole polygon	n	Coord. pts.
10	A	Hole polygon with spline curves	n	Coord. pts
245	F5	Image repeat request	2	Repeat count of 1-255 (1)
251-4	FB-FE	Load LUTs individually	258	Zero byte, one LUT
225	E1	Load selected LUT values	n	Target LUT, 1st location, # bytes, values
248	F8	Load three LUTs	770	Zero byte, three LUTs
211	D3	LUT compensation control	2	1-on; 2-no compensation
206	CE	Macro access mode	4	Access mode, font selection code
210	D2	Macro color control	2	1-internal colors; 2-specified colors
240	F0	Macro request	3	Family/font, macro code
232	E8	Macro rotation	5	Cos and sin of rotation angle (0()
222	DE	Macro string width	3	String width
6	06	Multiple edge	n	Color, coord. pts.
11	B	Multiple spline edge	n	Color, coord. pts
2	02	Non-filled arc	16	Color, width, 3 coord. pts.
0	00	Null command	1	---
255	FF	Null command	1	---
236	EC	Output device dependent	3	Start pixel X & Y coord. pts.
250	FA	Output image dimensions	10	Zero byte, 4 coordt. values
4	04	Polygon	n	Highlt, 0 byte, fill, coord. pts.

Table D-1 Alphabetical List of Opcodes by Opcode Description (continued...)

Dec (Continued...)	Hex	Opcode	Bytes	Contents
9	09	Polygon with spline curves	n	Color, coord. pts
130	82	Return macro string length	n	Mode bits
212	D4	RLC mapping data	n	N bytes of RLC image data
213	D5	RLC mapping parameters	14	Coord. pts., max. # of pixels/scan lines
219	DB	RLC merging box data	n	Zero byte, RLC data, x'00FF'
220	DC	RLC merging box dimensions	10	Zero byte, 2 X coords., 2 Y coords.
224	E0	SCODL control command	2	Subcommand byte
129	81	SCODL status report request	2	Subcommand byte to turn on/off status
223	DF	Select output device	3	Output device code, resolution
231	E7	Select string font	2	Family/font (default = 0)
226	E2	Set virtual screen image rotation	2	Rotation mode 0 to 3 (0(- 270()
1	01	Single edge (line)	10	Color, 2 coord. pts.
214	D6	Specify a kerning table	n	Font code, zero byte, N, 4-byte entries
218	DA	Specify background color	3	Zero byte, color code (0)
235	EB	Specify compressed mode wtg fcn	3	Zero byte, weighting function (0w8)
217	D9	Specify font attributes	7	Family/font, 0, X-base val, X-cap val
238	EE	Specify frame reference number	3	2-byte frame reference number
243	F3	Specify macro base address	5	X, Y coords of base address (0)
242	F2	Specify macro colors	3	Highlight color, fill color
244	F4	Specify macro scale	5	X, Y scale factors for macro coordts.
239	EF	Specify SCODL table sizes	11	5 values (8000/100/500/100/1024)
233	E9	Specify string mode and spacing	5	Spacing mode, text alignment pt., spacing
215	D7	Specify type of LUT compensation	n	Subcommand byte, code or option LUT
216	D8	Thicken edge	4	Subcommand, 2-byte thickening value
221	DD	Transparent color of RLC box	3	Zero byte, color code
249	F9	Virtual screen max dimensions	4	Subcommd byte, 1-2 coordt. values

Table D-1 Alphabetical List of Opcodes by Opcode Description

Index

Numbers in brackets refer to opcode descriptions listed in Section 4, *Description of Opcodes*

R

S

T

V

X

Y

Z